Hitachi Microcomputer

Development Environment System

E7000 SH7604 Emulator

User’s Manual

HITACHI

HS7604EPD70HE
ADE-702-105
Rev. 1.0
6/13/2000
Hitachi, Ltd.

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.

Preface

Thank you for purchasing the emulator for the Hitachi’s original microcomputer SH7604.

The emulator is an efficient software and hardware development tool for systems based on Hitachi’s original microcomputer SH7604 (hereafter abbreviated to SH-2). By exchanging the emulator pod, this emulator can also be used for other H-series microcomputers.

There are two types of SH-2 emulator: the E7000 and the E7000PC, which is used only with the IBM PC*[1]. This manual describes the functions and operating procedures of the E7000 and E7000PC for the SH-2 MCU.

To use the E7000, please read Part I, E7000 Guide, and Part III, Emulator Function Guide. To use the E7000PC, please read Part II, E7000PC Guide, and Part III, Emulator Function Guide.

Please read this manual carefully in order to gain a full understanding of the emulator’s performance. In particular, be sure to read section 1.1, Warnings, in Part I, E7000 Guide, or in Part II, E7000PC Guide before use.

Two system floppy disks are packaged together with the SH7604 emulator pod. Use the floppy disk corresponding to your emulator station, E7000 or E7000PC (the disks are labeled accordingly, as shown in the figure).

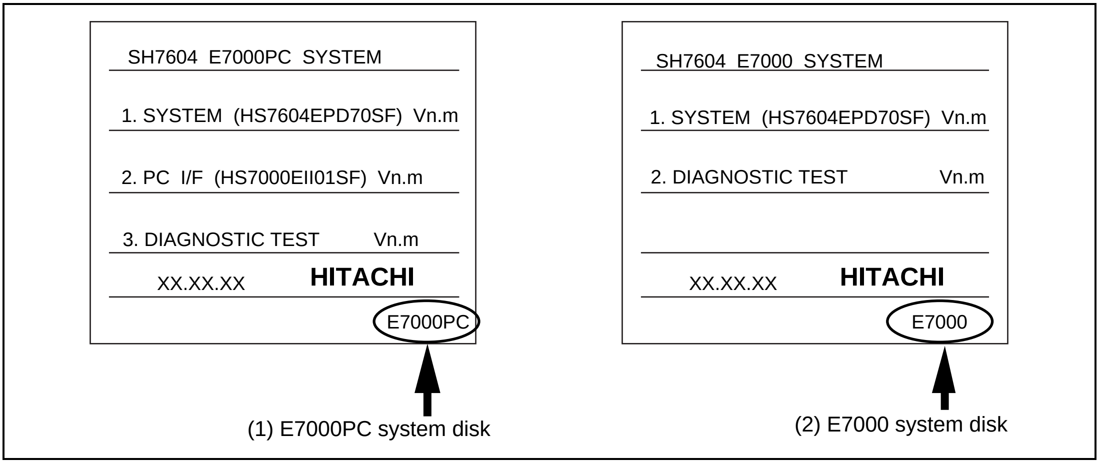


Figure System Disk Labels

Before using these system disks, back up or copy them as follows:

When Uusing the E7000 Emulator Station: Back up the system disk to a floppy disk using the E7000. For details on the back-up procedure, refer to the provided SH7604 Emulator User's Manual, section 3.6, Floppy Disk Backup in Part I, E7000 Guide.

When Using the E7000PC Emulator Station: Install (copy) the system disk to the personal computer connected to the E7000 PC. For details on the copy procedure, refer to the provided SH7604 Emulator User's Manual, section 3.4, System Disk Installation in Part II, E7000PC Guide.

Related Manuals:

SPARC*[2] SH-Series Cross Assembler User’s Manual
SPARC H-Series Linkage Editor User’s Manual
SPARC H-Series Librarian User’s Manual
SH E7000 Graphical Interface Software User’s Manual
SH-Series C Compiler User’s Manual
LAN Board Manual
HS7000EST01H Manual
HS7000ESTP1H Manual
Description Notes on Using the IBM PC Interface Board (HS7000EII01H)

When the E7000 is configured in remote mode, as described in section 2.3.2, RS-232C Interface System Configuration in Part I, E7000 Guide, refer to the following manual:

H-Series Interface Software User’s Manual

Notes: 1. IBM PC is a registered trademark of International Business Machines Corporation.
2. SPARC is a registered trademark of SPARC International, INC.

Contents

Part I E7000 Guide

Part II E7000PC Guide

Part III Emulator Function Guide

Figures

Part I E7000 Guide

Part II E7000PC Guide

Part III Emulator Function Guide

Tables

Part I E7000 Guide

Part II E7000PC Guide

Part III Emulator Function Guide

Part I   E7000 Guide

Section 1 Overview

This system is an efficient software and hardware development support tool for application systems using the SH7604 (abbreviated to SH-2) microcomputer developed by Hitachi, Ltd. The SH-2 MCU contains the following components on a single chip:

- High-speed CPU
- Cache
- Timers
- Serial communication interface
- Refresh controller
- DMAC

When the E7000 is connected to a user system, it operates in place of the SH-2 MCU and performs realtime emulation of the user system. Additionally, the E7000 provides functions for efficient software and hardware debugging.

The E7000 consists of an emulator station and emulator pod, as shown in figure 1.1. The emulator pod should be directly connected to the user system.

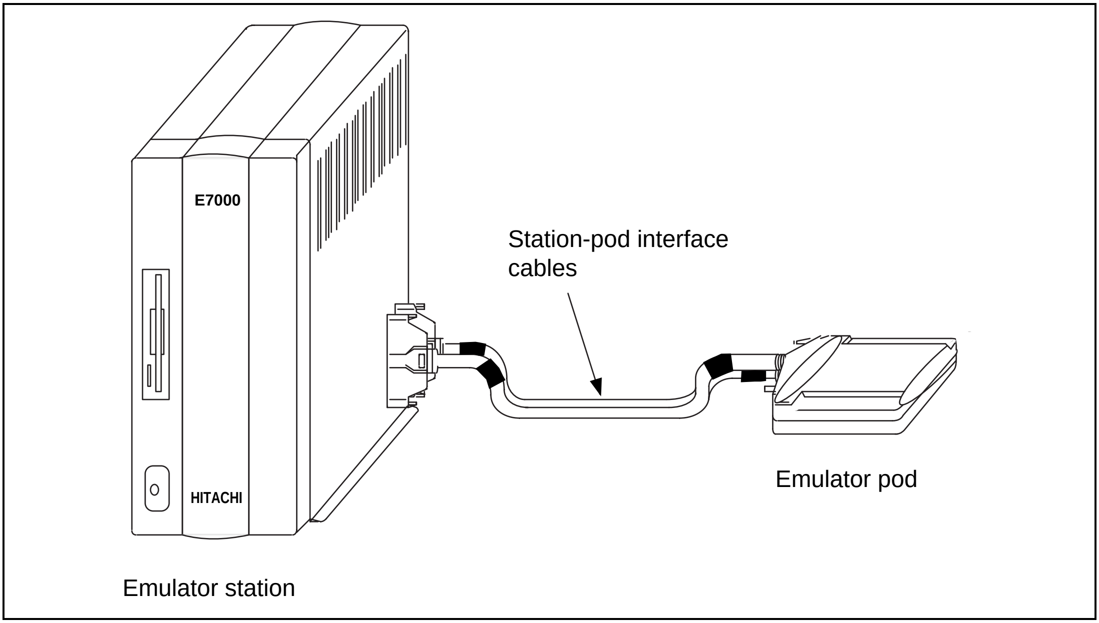


Figure 1.1 SH-2 E7000 Emulator

The E7000 provides the following features:

- Realtime emulation of SH-2
- A wide selection of emulation commands, promoting efficient system development
- Help functions to facilitate command usage without a manual
- Efficient debugging enabled by variable break functions and mass-storage trace memory (32 kcycles)
- Command execution during emulation, for example:
  - Trace data display
  - Emulation memory display and modification
- Measurement of subroutine execution time and frequency for evaluating the execution efficiency of user programs
- An optional LAN board for interfacing with workstations, enabling high-speed downloading (1 Mbyte/min) of user programs
- The LAN board contains Ethernet (10BASE5) and Cheapernet (10BASE2) interfaces. Ethernet is a registered trademark of the Xerox Corporation in the United States.
- E7000 graphical interface software (E7000 GUI: option) can be loaded into the workstation to enable:
  - Graphic display operations in a multi-window environment
  - Source level debugging
  - Graphic display of trace information
- Operation as a stand-alone system when connected to a console
- An RS-232C host system interface
- A Centronics printer interface
- A 3.5-inch floppy disk drive, which facilitates:
  - Loading, saving, and verifying user programs
  - Saving emulation results
  - Input and execution of commands using a floppy disk for external storage
- 512-kbyte emulation memory as substitute user system memory

1.1 Warnings

Before using the E7000, carefully read the following warnings. If the E7000 is not used correctly, breakdowns may occur.

Before System Initiation:

1. Check all components with the component list after unpacking the E7000.
2. Never place heavy objects on the casing.
3. Observe the following conditions in the area where the E7000 is to be used:
   - Make sure that the internal cooling fans on the sides of the emulator station are at least 20 cm (8") away from walls or other equipment.
   - Keep out of direct sunlight or heat. Refer to section 1.2, Environmental Conditions.
   - Use in an environment with constant temperature and humidity.
   - Protect the E7000 from dust.
   - Avoid subjecting the E7000 to excessive vibration. Refer to section 1.2, Environmental Conditions.
4. Protect the E7000 from excessive impacts and stresses.
5. Before using the E7000’s power supply, check its specifications such as power output, voltage, and frequency. For details on power supply, refer to section 1.2, Environmental Conditions.
6. When moving the E7000, take care not to vibrate or otherwise damage it. Pay special attention to exposed parts such as the power switch and I/O connectors.
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 E7000 and connected parts after connecting all cables. Cables should not be connected or removed when the power is on.
9. For details on differences between the SH-2 microcomputer and the E7000, refer to section 6, Differences between the SH-2 and the Emulator in Part III, Emulator Function Guide.
10. The optional 1-Mbyte or 4-Mbyte emulation memory board and the bus monitor board for the E7000 cannot be used in the E7000. Check that neither the emulation board nor the bus monitor board is installed in the E7000 emulator station. If installed, remove the boards from the emulator station before connecting the emulator pod.

Floppy Disk:

If the floppy disk is removed when a file is being accessed, the file may be damaged and processing may fail.

1.2 Environmental Conditions

Observe the conditions listed in table 1.1 when using the E7000.

Table 1.1 Environmental Conditions

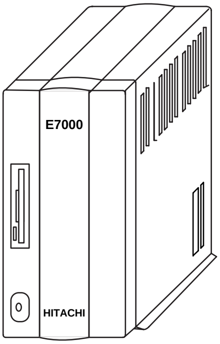
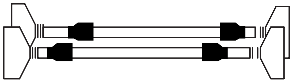
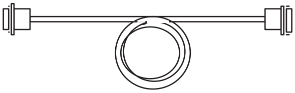
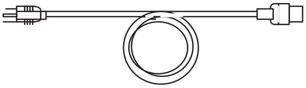
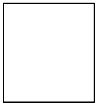
Item	Specifications
Temperature	Operating: +10 to +35°C Storage: –10 to +50°C
Humidity	Operating: 35 to 80% RH (no condensation) Storage: 35 to 80% RH (no condensation)
Vibration	Operating: 2.45 m/s^2 max. Storage: 4.9 m/s^2 max. Transportation: 14.7 m/s^2 max.
AC input power	Voltage: 100/200 VAC ±10% Frequency: 50/60 Hz Power consumption: 200 VA
Ambient gases	Must be no corrosive gases

1.3 Components

The E7000 consists of the emulator station and emulator pod. Check all the components after unpacking.

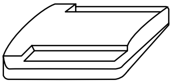
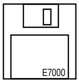
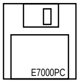
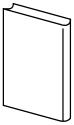
1.3.1 E7000 Emulator Station

Table 1.2 E7000 Emulator Station Components

Item	Configuration		Quantity	Remarks
Hardware	Emulator station		1	Power supply, 3.5-inch floppy disk drive, control board, and trace board
	Station-pod interface cables		2	50 cm
	Console interface cable		1	3 m RS-232C
	AC power cable			
	Fuse		1	Spare (3 A)
Documen- tation	HS7000EST01H Description Notes		1	HS7000EST01HE

1.3.2 E7000 Emulator Pod

Table 1.3 E7000 Emulator Pod Components

Item	Configuration		Quantity	Remarks
Hardware	Emulator pod		1	Fitted with two boards
Software	Floppy disks		1	E7000 system program
			1	E7000PC system program (cannot be used with the E7000)
Documentation	SH7604 E7000 Emulator User's Manual		1	HS7604EPD70HE

1.3.3 Options

In addition to the emulator station and pod components, the options listed in table 1.4 are also available. Refer to each option manual for details on these optional components.

Table 1.4 Optional Component Specifications

Item	Model Name	Specifications
LAN board	HS7000ELN01H	• TCP/IP communications protocol • Ethernet (10BASE5) • Cheapernet (10BASE2)
Host system interface cable	HS7000EHT71H	RS-232C interface
Printer cable	HS7000EPR71H	Centronics interface

2.1 E7000 Hardware Components

As shown in figure 2.1, the E7000 consists of an emulator station (including two RS-232C interface cables and a printer cable) and emulator pod. By installing a LAN board (option), the emulator station can be connected to a workstation.

Note: Optional emulation memory boards and bus monitor boards cannot be installed in the E7000.

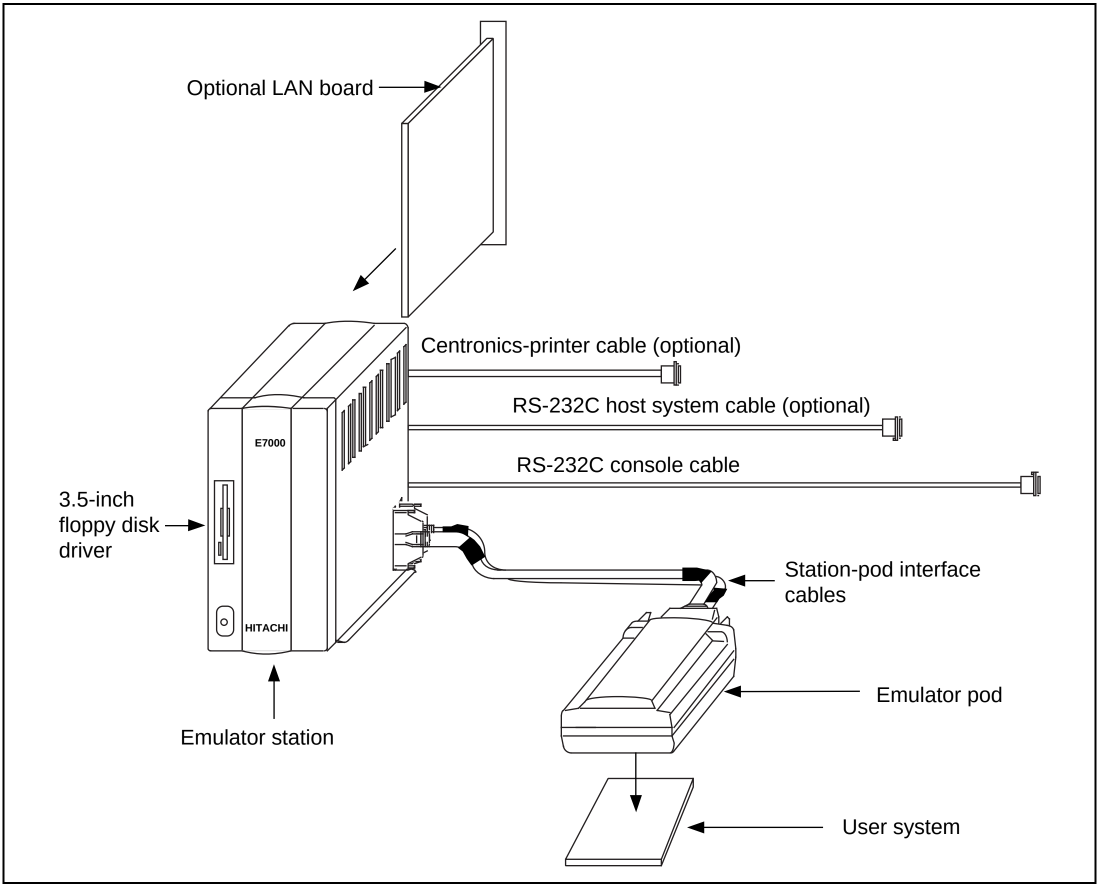

Optional LAN board
Centronics-printer cable (optional)
RS-232C host system cable (optional)
RS-232C console cable
E7000
3.5-inch floppy disk driver
HITACHI
Station-pod interface cables
Emulator pod
Emulator station
User system

Figure 2.1 E7000 Emulator Hardware Components

2.1.1 Emulator Station Components

Front Panel:

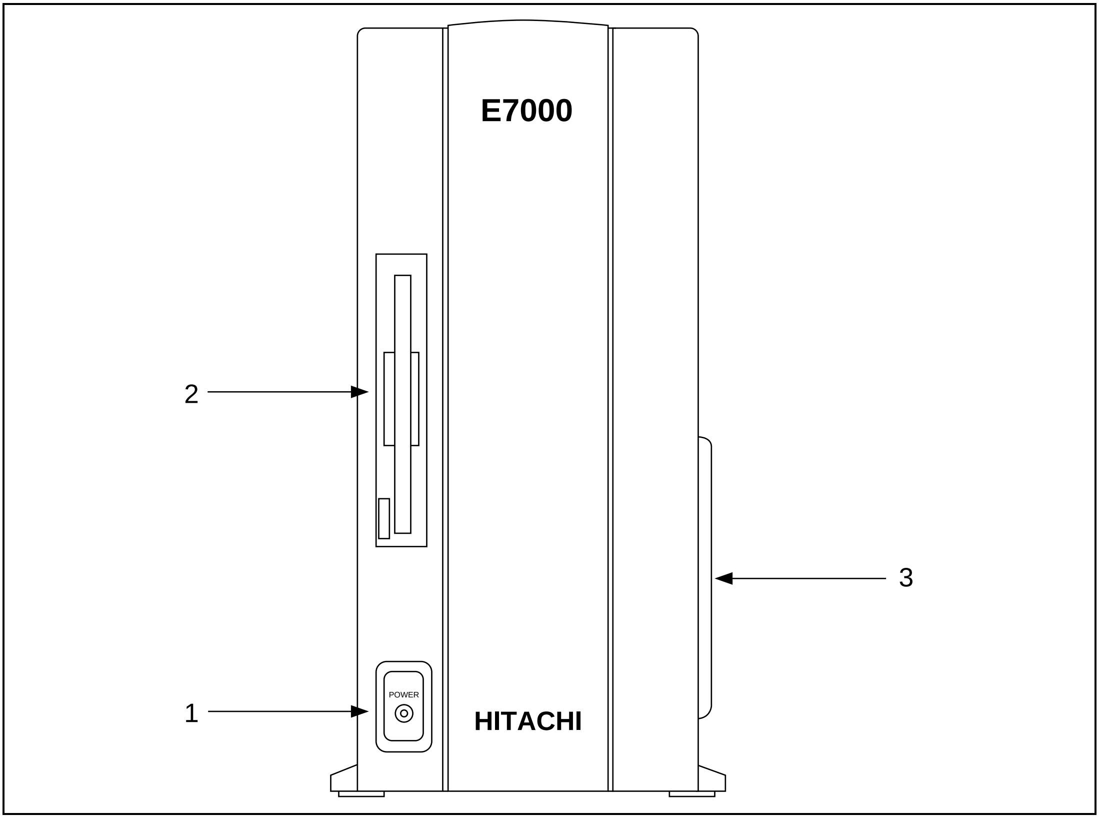


Figure 2.2 E7000 Emulator Station Front Panel

1. Power lamp

   Lights when the E7000 power is on.

2. 3.5-inch floppy disk drive

   For loading the E7000 system program, as well as loading, saving, and verifying the contents of the user system memory.

3. Station-pod interface cable connectors

   For connecting the emulator pod to the emulator station.

Rear Panel:

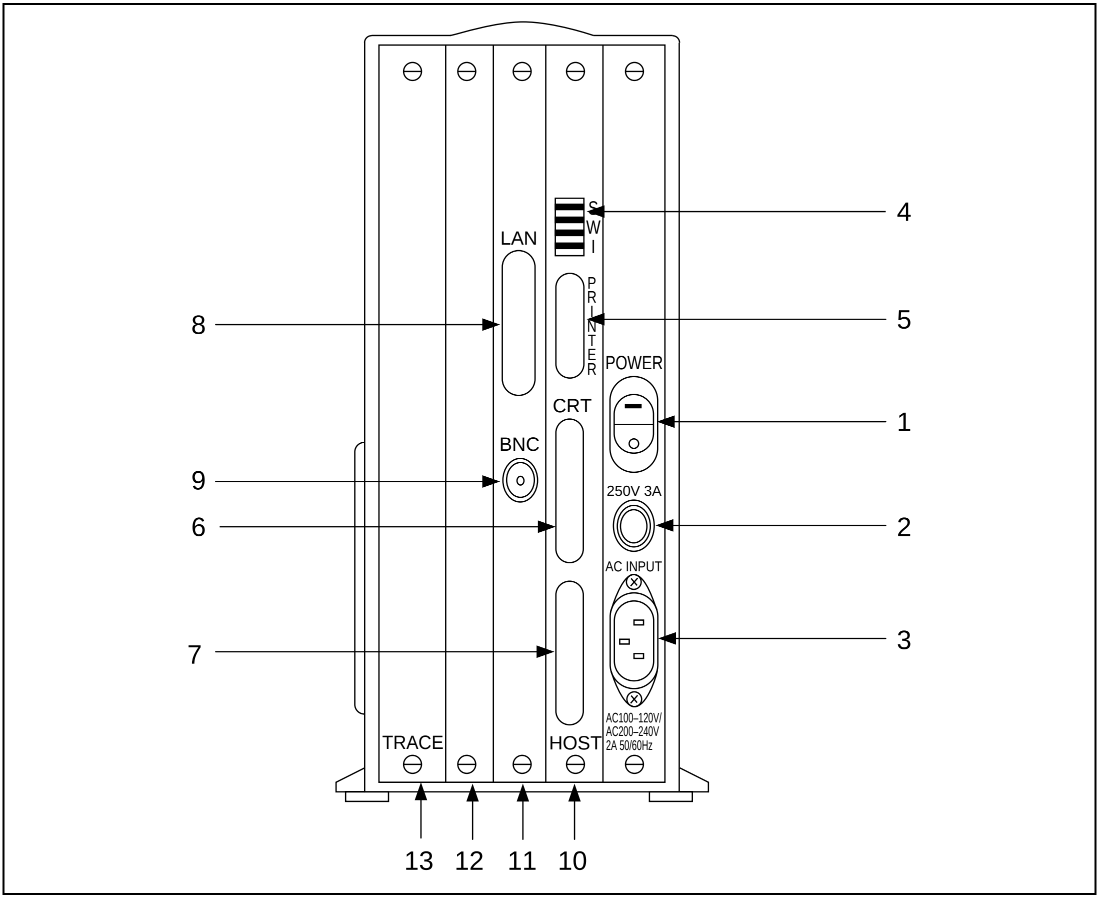


Figure 2.3 E7000 Emulator Station Rear Panel

1. Power switch

   Turning this switch to I (input) supplies power to the E7000 (emulator station and pod).

2. Fuse box

   Contains a 3-A 250 VAC fuse.

3. AC power connector

   For an 100/200 VAC power supply.

4. Console interface switch

   For changing the transfer speed, data bit length, stop-bit length, parity specifications, and LAN interface settings when interfacing with a console. Marked SW1.

5. Printer connector

   For a printer conforming to Centronics specifications. Marked PRINTER.

6. Console connector

   For an RS-232C console. Marked CRT.

7. Host system connector

   For RS-232C communication with a host system. Marked HOST.

8. Ethernet connector

   For an Ethernet cable. Marked LAN.

9. Cheapernet connector

   For a Cheapernet cable. Marked BNC.

10. Control board slot

    For installing the control board.

11. LAN-board slot

    For installing the optional LAN board.

12. Extension slot

    For system extension.

13. Trace board slot

    For installing the trace board.

2.1.2 Emulator Pod Components

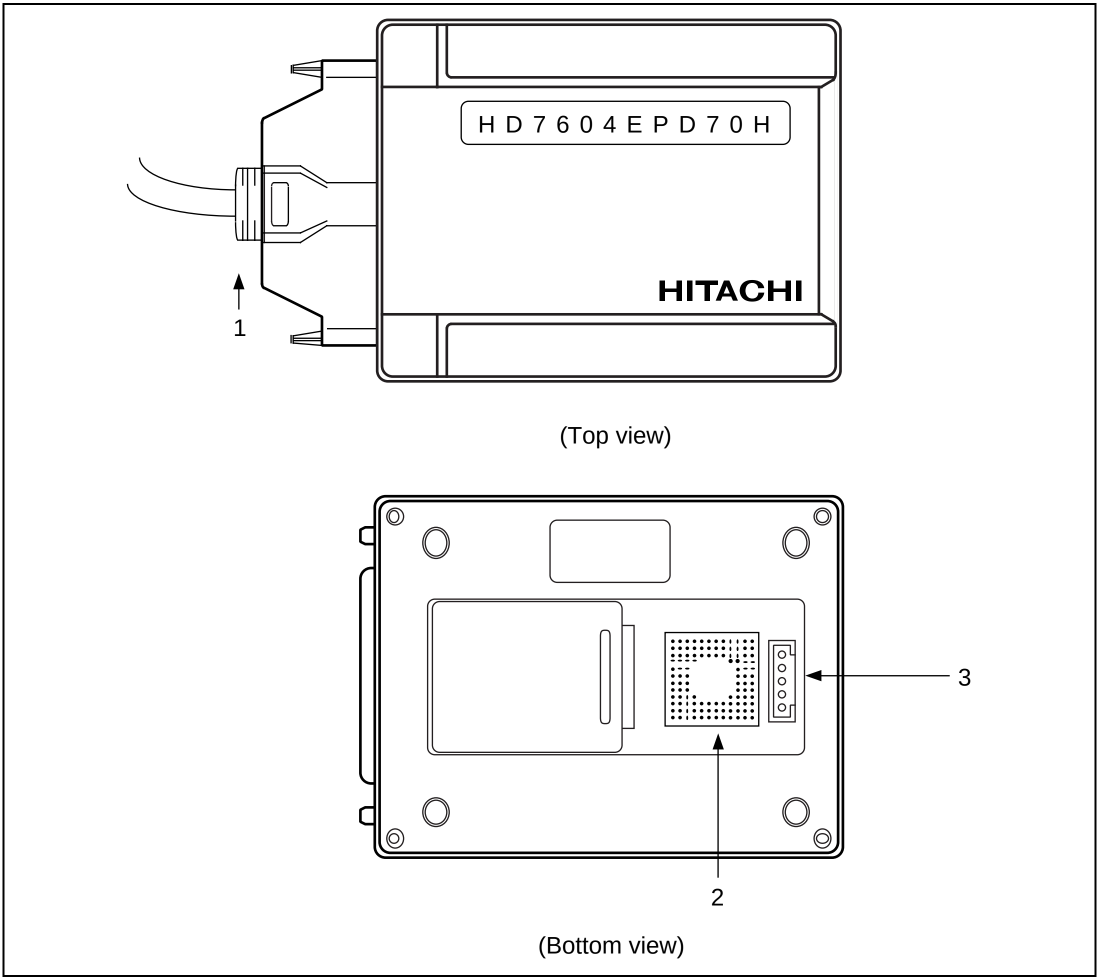


Figure 2.4 E7000 Emulator Pod

1. Station-pod interface cables

   For connecting the emulator station to the emulator pod.

2. User system connector

   For connection of the user system.

3. External probe connector

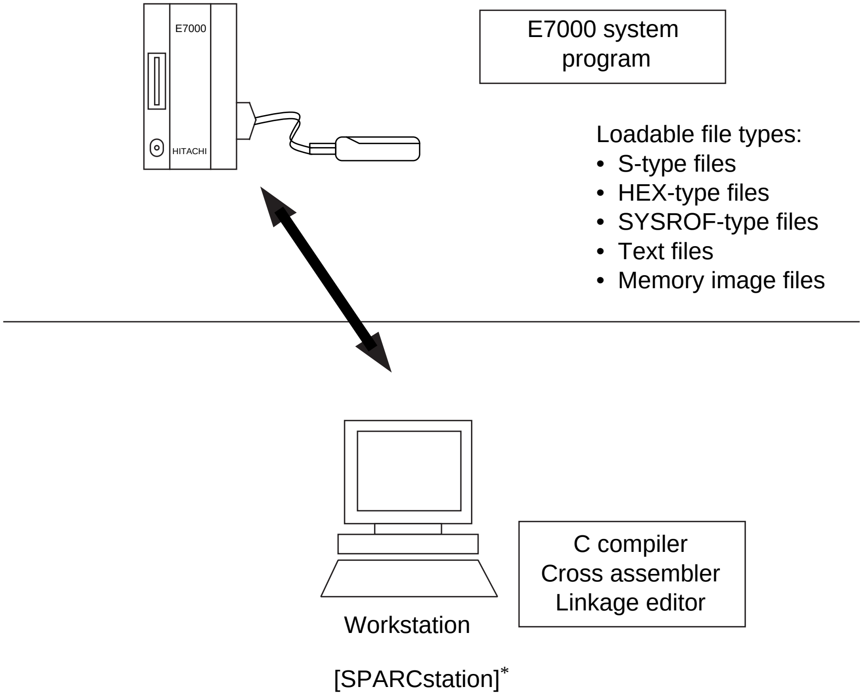
   For connection of the external probe.

2.2 E7000 Software Components

The E7000's software components are illustrated in figure 2.5. The emulator pod contains two 3.5-inch floppy disks: the E7000 emulator system disk has "E7000" written under "HITACHI" on its label. The system disk files are described in table 2.1.

Table 2.1 Contents of E7000 System Disk

File Name	Contents	Description
E7000.SYS	E7000 system program	Controls the emulator pod and processes commands, such as emulation commands. Loaded into the E7000 memory after the E7000 system program is activated.
SHPOD764.SYS	SH-2 control program	Controls the MCU within the emulator pod. Loaded into the E7000 memory after the E7000 system program is activated.
SHCNF764.SYS	Configuration file	Contains MCU operating mode and MAP information. Loaded with the E7000 system program.
LANCNF.SYS	LAN configuration file	Stores the host system name and IP address information when the E7000 is connected to a workstation by a LAN interface.
DIAG.TM	Diagnostic program	Loaded into the emulator station memory for testing and maintenance.



Note: SPARCstation is a registered trademark of SPARC International, INC. (United States) and has permission of monopolistic use granted by Sun Microsystems Corporation (United States).

Figure 2.5 E7000 Emulator Software Components

2.3 System Configuration

The E7000 can be connected with the host system via a LAN or an RS-232C interface.

2.3.1 System Configuration Using LAN Interface

By installing an optional LAN board in the emulator station, the E7000 can be connected to a workstation through a LAN interface. The LAN interface employs the TCP/IP protocol and the LAN board contains connectors for both Cheapernet (10BASE2) and Ethernet (10BASE5). The system configuration using a LAN interface is shown in figure 2.6.

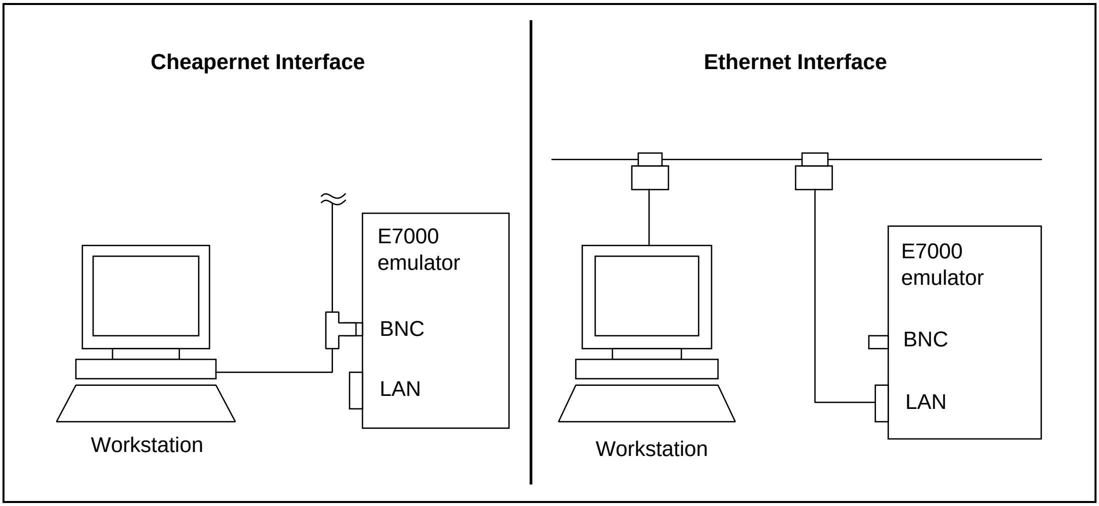


Figure 2.6 System Configuration Using LAN Interface

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

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

2.3.2 System Configuration Using RS-232C Interface

Using an RS-232C interface, the E7000 system can be configured in any of the ways shown in figure 2.7.

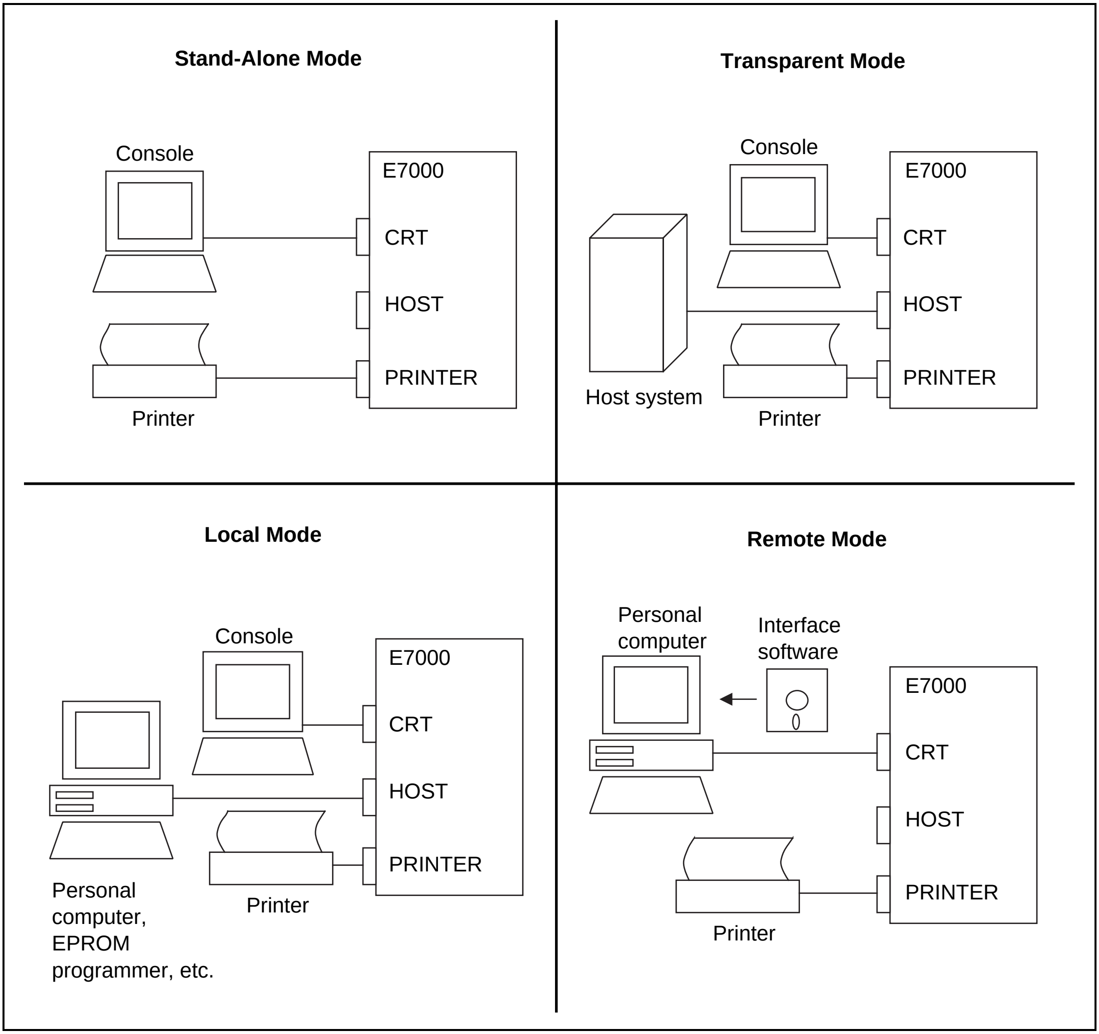


Figure 2.7 System Configuration Using RS-232C Interface

Stand-Alone Mode: Configuration in which the E7000 is connected to the console and operates alone.

Transparent Mode: Configuration in which the console connected to the E7000 can also serve as a console for the host system. The console allocation is switched using the TERMINAL command.

Local Mode: Configuration which allows data transfer between the E7000 and a personal computer. In this mode, data can be sent using the standard commands of the personal computer. This configuration can also be used to connect to an EPROM programmer.

Remote Mode: Configuration in which a personal computer can be used as the console or the host machine for data transfer. Interface software must be loaded in the personal computer.

Section 3 Preparation before Use

3.1 E7000 Preparation

Unpack the E7000 and prepare it for use as follows:

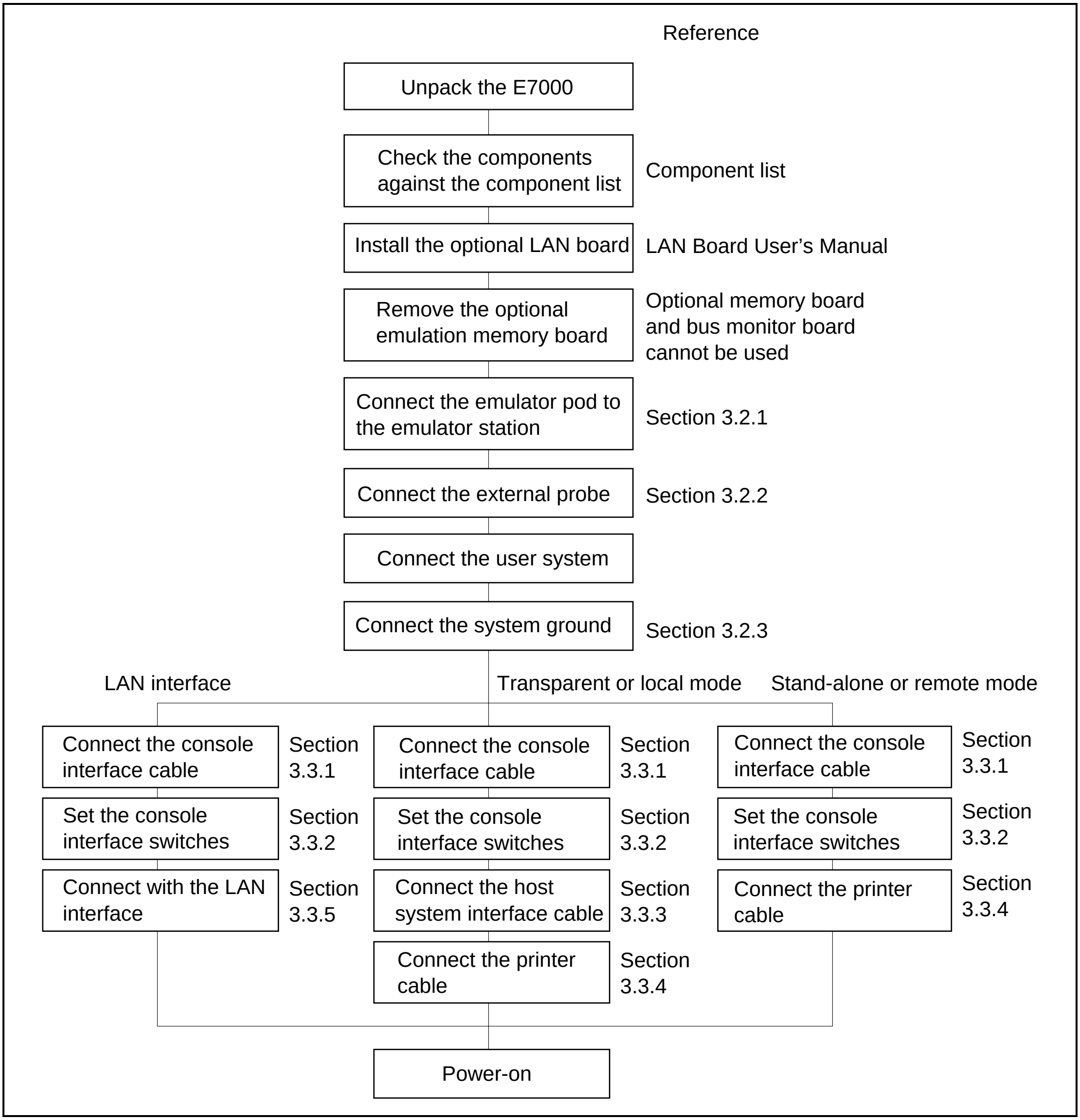

Reference
Unpack the E7000
Check the components against the component list
Component list
Install the optional LAN board
LAN Board User's Manual
Remove the optional emulation memory board
Optional memory board and bus monitor board cannot be used
Connect the emulator pod to the emulator station
Section 3.2.1
Connect the external probe
Section 3.2.2
Connect the user system
Connect the system ground
Section 3.2.3
LAN interface
Transparent or local mode
Stand-alone or remote mode
Connect the console interface cable
Section 3.3.1
Set the console interface switches
Section 3.3.2
Connect with the LAN interface
Section 3.3.5
Connect the console interface cable
Section 3.3.1
Set the console interface switches
Section 3.3.2
Connect the host system interface cable
Section 3.3.3
Connect the printer cable
Section 3.3.4
Connect the console interface cable
Section 3.3.1
Set the console interface switches
Section 3.3.2
Connect the printer cable
Section 3.3.4
Power-on

Figure 3.1 E7000 Preparation Flow Chart

3.2 E7000 Connection

3.2.1 Connecting Emulator Pod

The emulator pod and the emulator station are packed separately. Use the following procedure to connect the emulator pod to the emulator station, or to disconnect it when moving the E7000:

(1) Check that the E7000 power is off by ensuring that the power lamp on the left side of the emulator station front panel is extinguished.

(2) Remove the AC power cable for the emulator station from the outlet.

(3) Connect station-pod interface cables P1 and P2 to station-pod interface connectors J1 and J2 on the right side of the emulator station, respectively. Insert the longer screw of each cable to the connector screw hole without a spacer, and the shorter screw to the hole with a spacer. Tighten the longer screw first until the shorter screw reaches the spacer, then alternately tighten the longer and shorter screws. Figure 3.2 shows how to connect the station-pod interface cables to the emulator station.

Note: When connecting the cables, prevent the upper or lower side of the cables from lifting off the connector. Tighten the screws and push the cables gradually toward the connector.

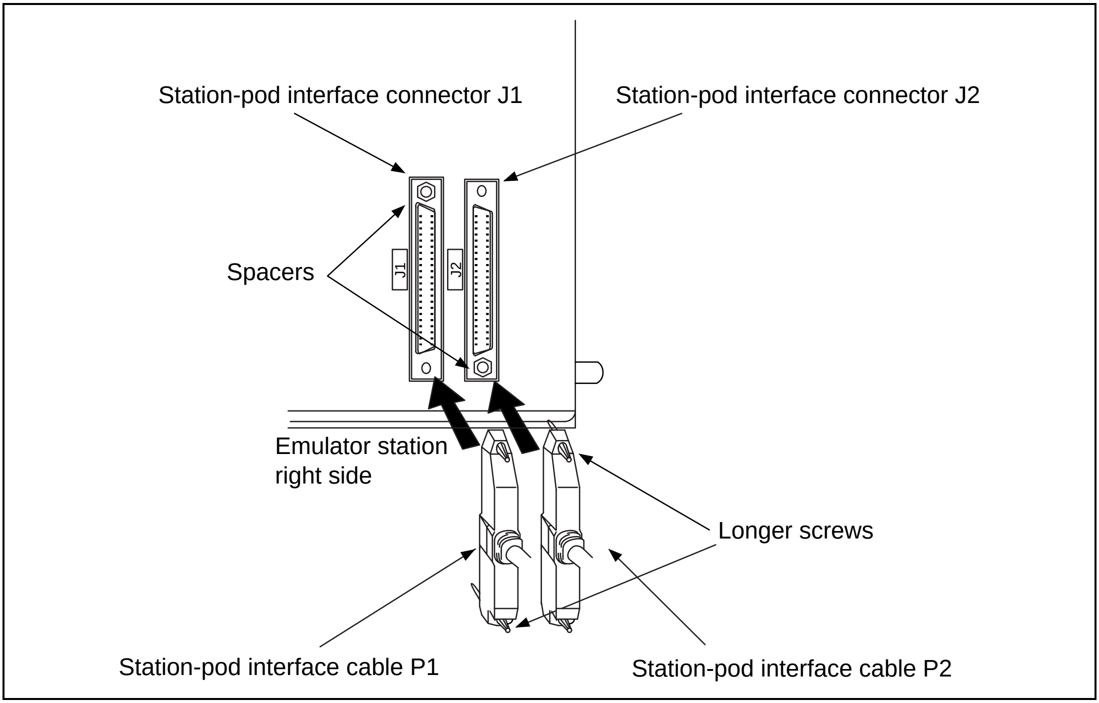

Station-pod interface connector J1
Station-pod interface connector J2
Spacers
J1
J2
Emulator station right side
Longer screws
Station-pod interface cable P1
Station-pod interface cable P2

Figure 3.2 Connecting Station-Pod Interface Cables to Emulator Station

(4) Connect the station-pod interface cables P1 and P2 to station-pod interface connectors J1 and J2, respectively, in the same way as connection to the emulator station. Tighten the screws in the same way as in step (3). See figure 3.3 for details.

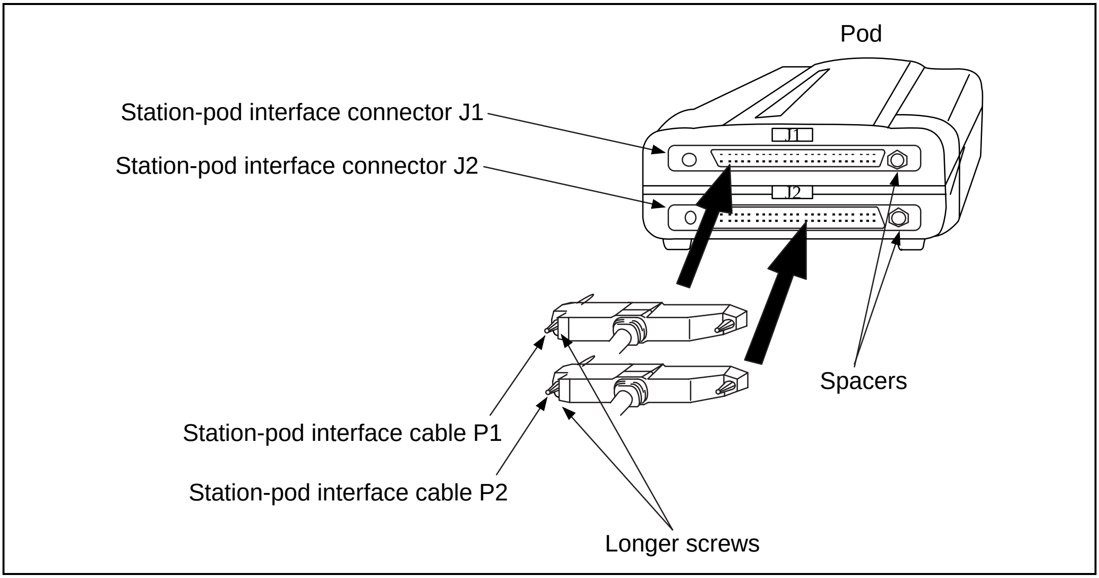
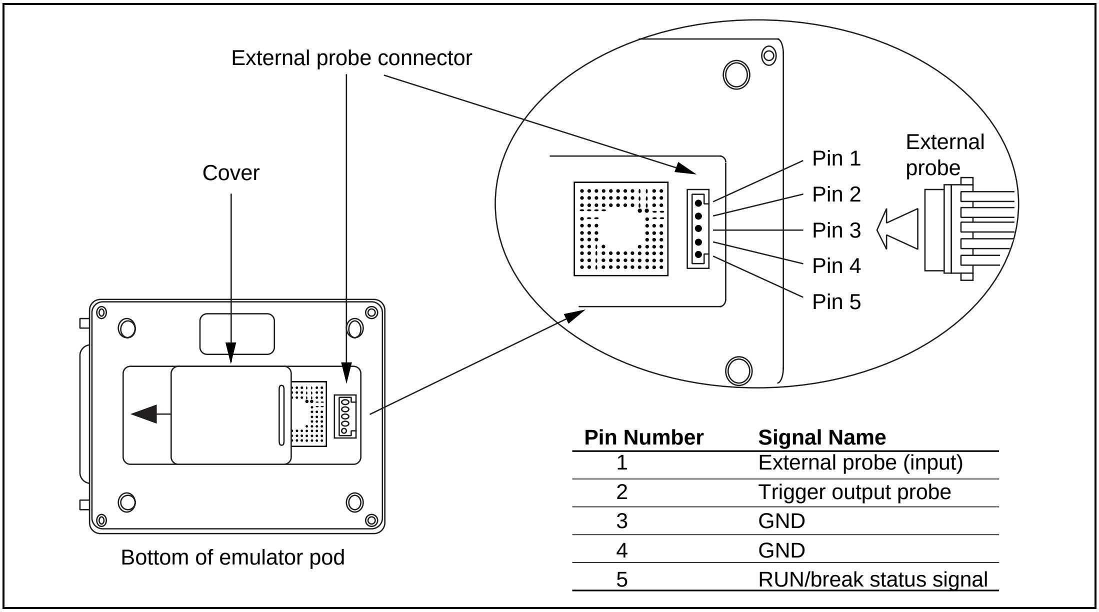

Pod
Station-pod interface connector J1
Station-pod interface connector J2
J1
J2
Spacers
Station-pod interface cable P1
Station-pod interface cable P2
Longer screws

Figure 3.3 Connecting Station-Pod Interface Cables to Emulator Pod

3.2.2 External Probe Connector

When an external probe is connected to the emulator pod, it enables external signal trace and multibreak detection. Figure 3.4 shows the external probe connector.



Pin Number	Signal Name
1	External probe (input)
2	Trigger output probe
3	GND
4	GND
5	RUN/break status signal

Figure 3.4 Connecting External Probe

3.2.3 Connecting System Ground

The E7000's signal ground is connected to the user system's signal ground via the emulator pod. In the emulator station, the signal ground and frame ground are connected (figure 3.5). At the user system, connect the frame ground only; do not connect the signal ground to the frame ground. If it is difficult to separate the signal ground from the frame ground, ground the user system at the same outlet as the E7000's power supply (figure 3.6).

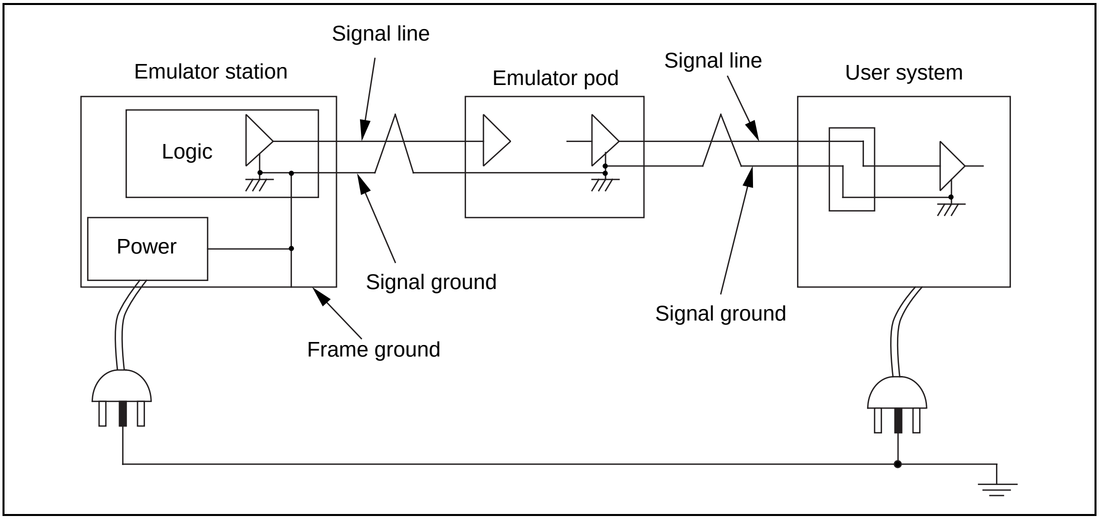


Figure 3.5 Connecting System Ground

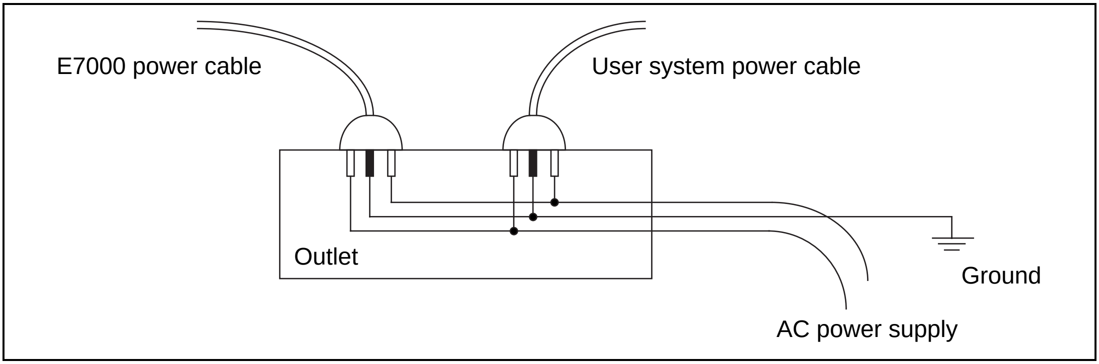


Figure 3.6 Connecting Frame Ground

The user system must be connected to an appropriate ground so as to minimize noise, ground loops, and other adverse effects. Confirm that the ground pins of the user system interface cable are firmly connected to the user system's ground.

3.3 System Connection

This section describes how to connect the E7000 to a workstation, personal computer, console, and printer. Connectors for each of these are located on the emulator station as shown in figure 2.3.

3.3.1 Connecting to a Console

The console connector (marked CRT) located on the emulator station rear panel conforms to the RS-232C specifications (table 3.1). A console can be connected to this by the console interface cable supplied with the E7000, making it possible to input commands and check their results on the console. This connection is also used to specify the IP address before connecting a workstation via the LAN interface.

Table 3.1 Console Interface Specifications

Item	Specification
Signal level	RS-232C High: +5 to +15 V Low: –5 to –15 V
Transfer rate	2400/4800/9600/19200 BPS
Synchronization method	Asynchronous method
Start bits	1 bit
Data bits	7/8 bits
Stop bits	1/2 bits
Parity	Even/odd or none
Control method	X-ON/X-OFF control (Refer to 9.3.1, Control Methods in Part III, Emulator Function Guide.)

BPS: Bits per second

For the console connector pin assignment and signal names, refer to section B.1, Console Connector. For console interface cable connection, refer to section B.5, Console Interface Cable Connection.

3.3.2 Setting up Console Interface

The transfer rate, number of data bits, number of stop bits, and parity can be changed using the console interface switches (SW1) on the emulator station rear panel. One of these switches selects either the console interface or LAN interface.

Eight console interface switches (S1 to S8) are arranged as shown in figure 3.7. When a switch is pushed to the right, it turns on, and when it is pushed to the left, it turns off.

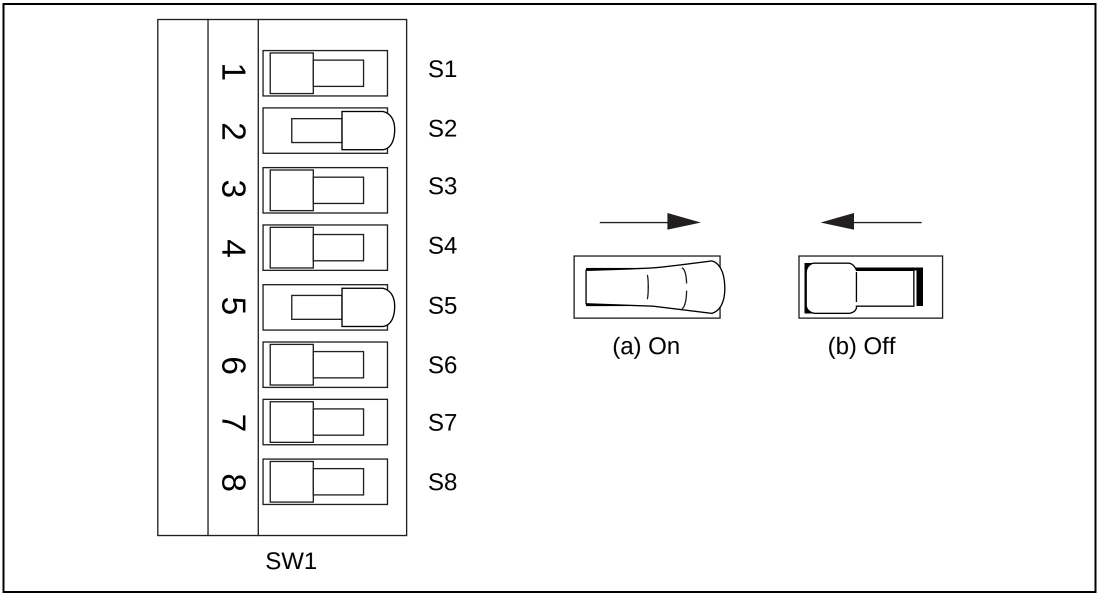


Figure 3.7 Console Interface Switches

The console interface settings are changed by turning these switches on or off as shown in table 3.2.

Table 3.2 Console Interface Switch Settings

Transfer rate

Transfer Rate	S1	S2	S3
19200 BPS	On	On	Off
9600 BPS	Off	On	Off*
4800 BPS	On	Off	Off
2400 BPS	Off	Off	Off

Number of stop bits

Stop Bits	S4
1 bit	Off*
2 bits	On

Number of data bits

Data Bits	S5
7 bits	Off
8 bits	On*

Parity

Parity	S6
None	Off*
Even/odd	On

Even/odd parity (only valid if parity switch is on)

Parity	S7
Even	Off*
Odd	On

Console/LAN interface selection

Interface	S8
Console	Off*
LAN	On

Note: * indicates setting at shipment.

3.3.3 Connecting to a Host System

This section describes how to set the host system interface when the E7000 is connected to a host system such as a personal computer or EPROM programmer.

The host system connector (marked HOST) located on the emulator station rear panel conforms to the RS-232C specifications (table 3.3). Connecting a host system to this connector enables data transfer between the E7000 and the host system.

Table 3.3 Host System 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 BPS
Synchronization method	Asynchronous method
Start bits	1 bit
Data bits	7/8 bits
Stop bits	1/2 bits
Parity	Even/odd or none
Control method	X-ON/X-OFF control, RTS/CTS control

BPS: Bits per second

Host System Interface Settings at E7000 Start-up: When the E7000 is turned on, or when the E7000 system program is initiated, the host system interface settings are determined by the console interface switches in the same way as the console interface (control method will be X-ON/X-OFF control). For details, refer to section 9.3.1, Control Methods in Part III, Emulator Function Guide.

Changing Host System Interface Settings: The transfer rate, number of data bits, number of stop bits, parity, and control method can be changed with the HOST command. For details, refer to section 9.4.1, HOST in Part III, Emulator Function Guide.

For the host system connector pin assignments and signal names, refer to section B.2, Host System Connector. For connection of optional host system interface cable, refer to section B.6, Host System Interface Cable Connection.

3.3.4 Connecting to a Printer

The printer connector (marked PRINTER) is located on the emulator station rear panel. Connecting a printer to this connector enables the command execution results to be printed. The printer interface conforms to the Centronics specifications.

For the printer connector pin assignments and signal names, refer to section B.3, Printer Connector. For connection of the optional printer cable, refer to section B.7, Printer Cable Connection.

3.3.5 Connecting to a LAN Interface

The LAN board for the E7000 supports Ethernet (10BASE5) and Cheapernet (10BASE2) interfaces conforming to Ethernet specifications V.2.0.

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 3.4 shows the Ethernet and Cheapernet specifications.

Table 3.4 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 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 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 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.
  - 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 E7000 to the workstation.
- UDP (User Diagram Protocol)
  Not supported.

Session, Presentation, and Application Layers:

- FTP (File Transfer Protocol)
  The E7000 operates as a client.
- TELNET (Teletype Network)
  The E7000 operates as a server.

Note: The E7000 does not communicate through routers or gateways.

3.3.6 System Connection Examples

Some examples of system configuration are shown below.

Ethernet Interface: The LAN board has a 15-pin D-SUB connector for the Ethernet transceiver cable. Figure 3.8 shows an example of Ethernet system configuration. Use commercially available Ethernet transceivers and transceiver cables. Table 3.5 shows a recommended transceiver and transceiver cable.

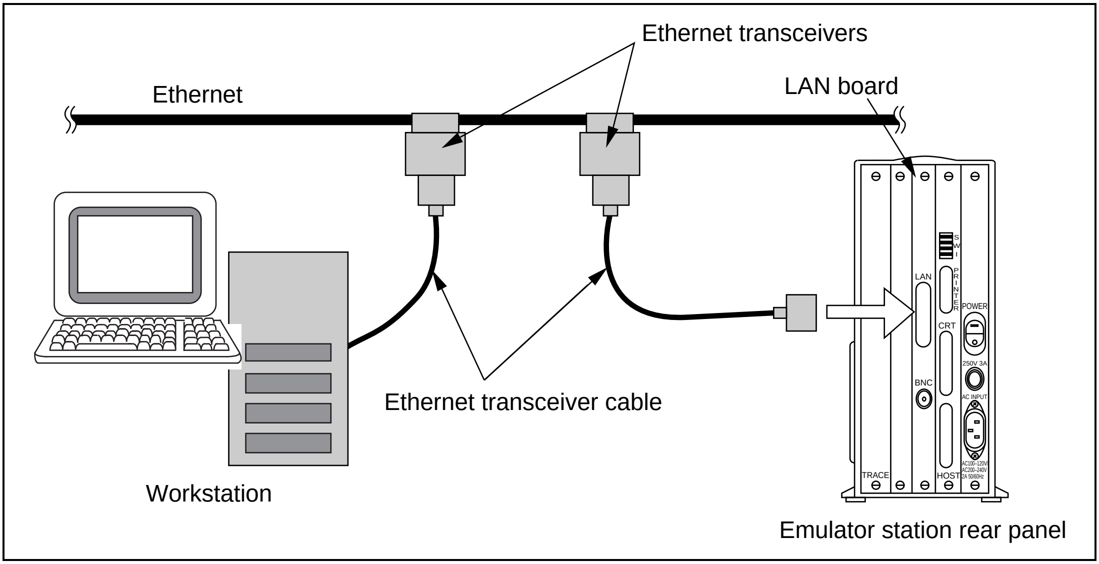


Figure 3.8 Ethernet Interface

Table 3.5 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 Ethernet, refer to the LAN board user’s manual.

Cheapernet Interface: The LAN board of the E7000 has a transceiver and a BNC connector for Cheapernet interface. Figure 3.9 shows an example of Cheapernet system configuration. Use a commercially available Cheapernet BNC T-type connector with a characteristic impedance of 50 Ω and an RG-58A/U thin-wire cable or its equivalent. Table 3.6 shows a recommended BNC T-type connector and thin-wire cable.

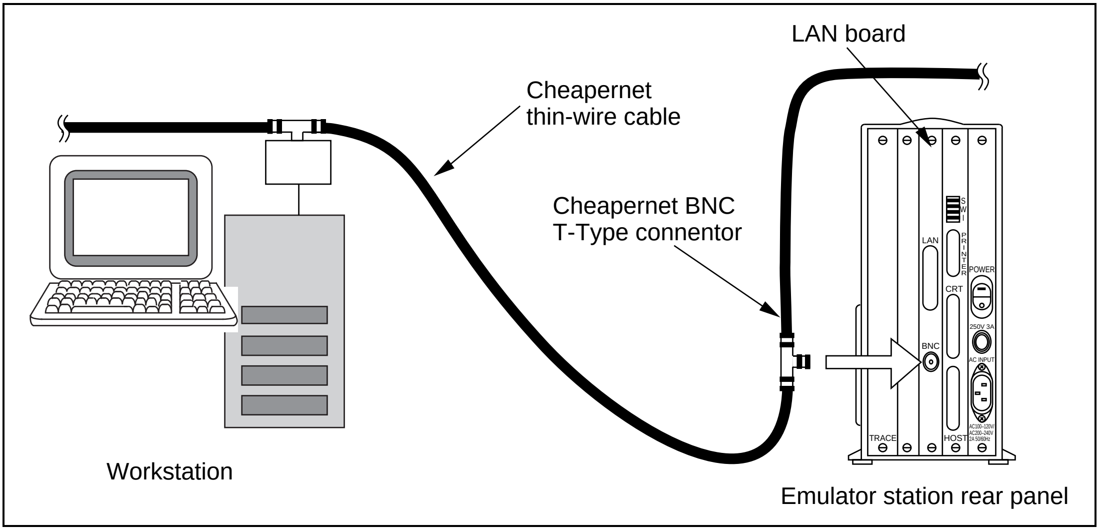


Figure 3.9 Cheapernet Interface

Table 3.6 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 user’s manual.

Stand-Alone Mode: A console is connected to the E7000 as shown in figure 3.10.

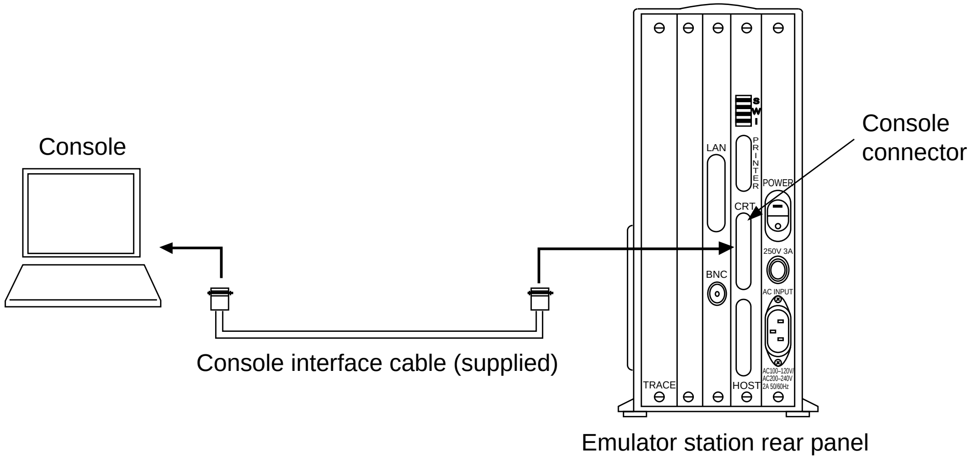


Figure 3.10 Connection in Stand-Alone Mode

Transparent Mode: A console and host system are connected to the E7000 as shown in figure 3.11.

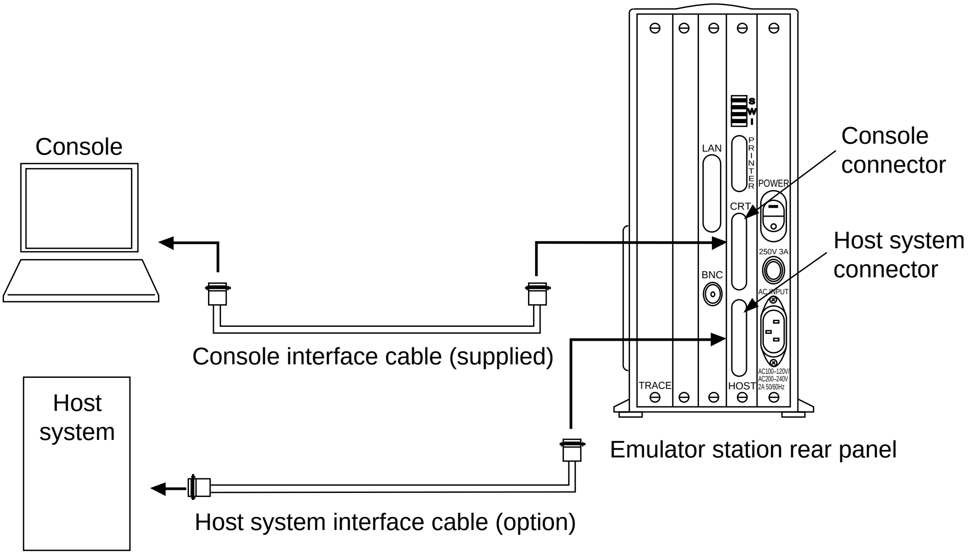


Figure 3.11 Connection in Transparent Mode

Local Mode: A console and host system are connected to the E7000 as shown in figure 3.12.

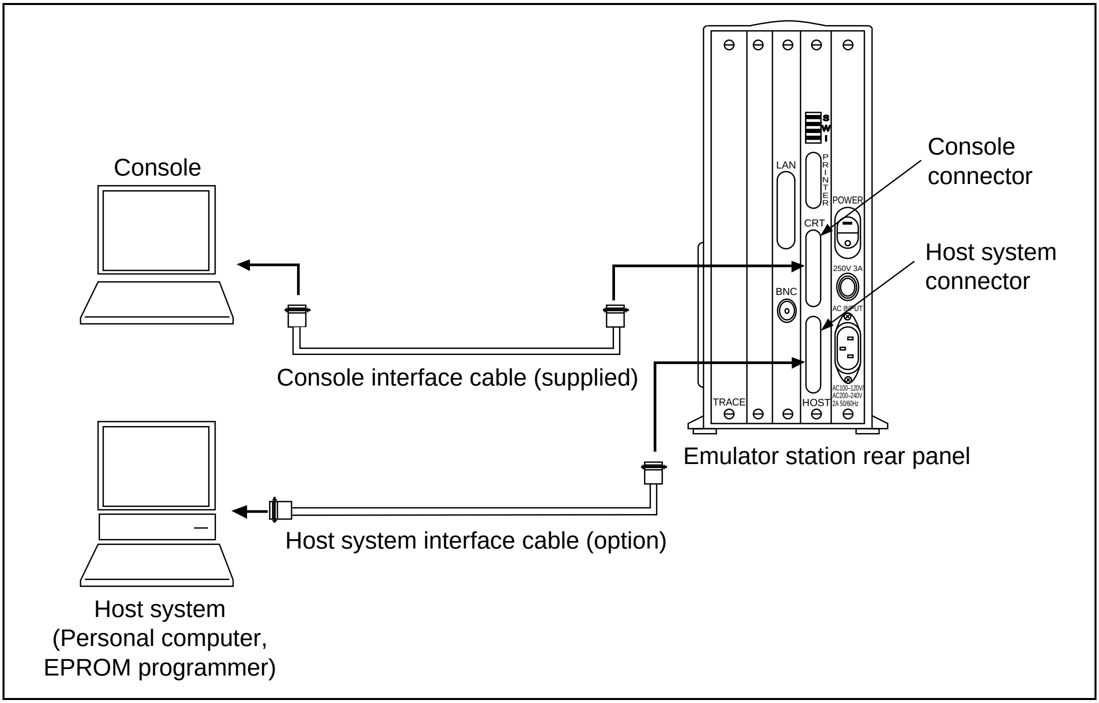

Console
Console connector
Host system connector
LAN
PRINTER
SWI
POWER
CRT
250V 3A
BNC
AC INPUT
AC100–120V/ AC200–240V 2A 50/60Hz
TRACE
HOST
Console interface cable (supplied)
Emulator station rear panel
Host system interface cable (option)
Host system (Personal computer, EPROM programmer)

Figure 3.12 Connection in Local Mode

Remote Mode: A host system is connected to the E7000 as shown in figure 3.13. The control method in remote mode is X-ON/X-OFF.

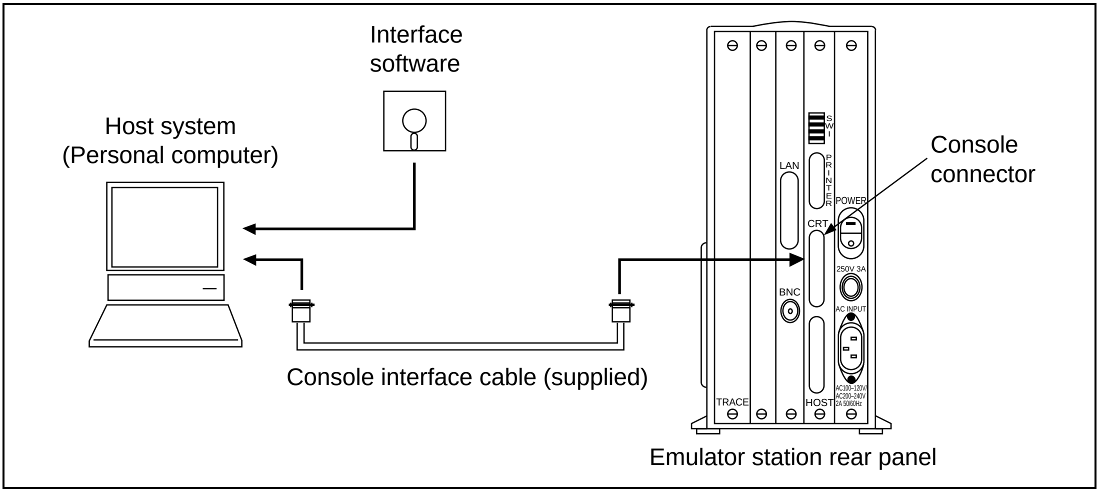

Interface software
Host system (Personal computer)
Console connector
LAN
PRINTER
SWI
POWER
CRT
250V 3A
BNC
AC INPUT
AC100–120V/ AC200–240V 2A 50/60Hz
TRACE
HOST
Console interface cable (supplied)
Emulator station rear panel

Figure 3.13 Connection in Remote Mode

Printer: A printer is connected to the E7000 as shown in figure 3.14.

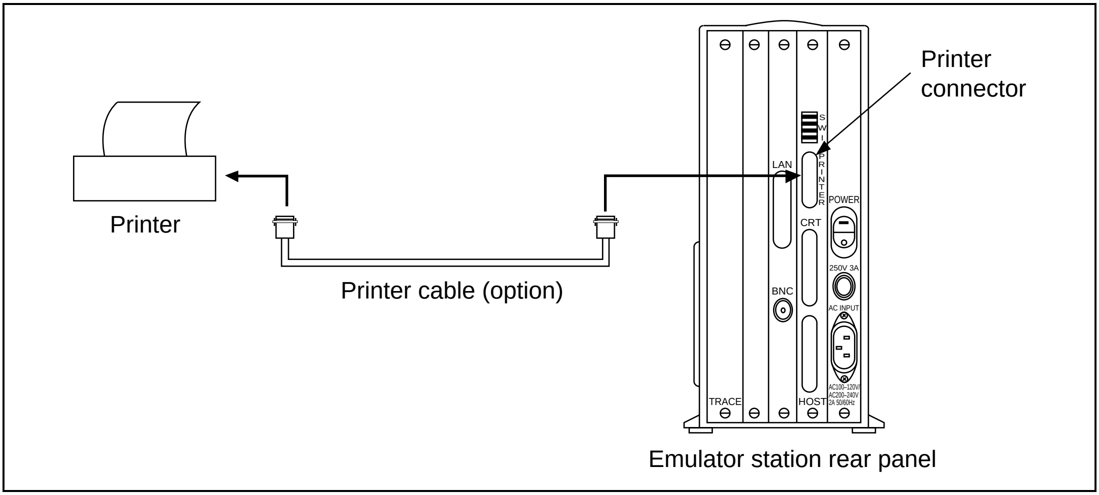


Figure 3.14 Printer Connection

3.4 Power-On

The E7000 power-on procedure differs in each system configuration. Power on the E7000 in the appropriate way for the system configuration, as shown below.

3.4.1 Power-On Procedure for LAN Interface

Figure 3.15 shows the power-on procedure when the LAN interface is used.

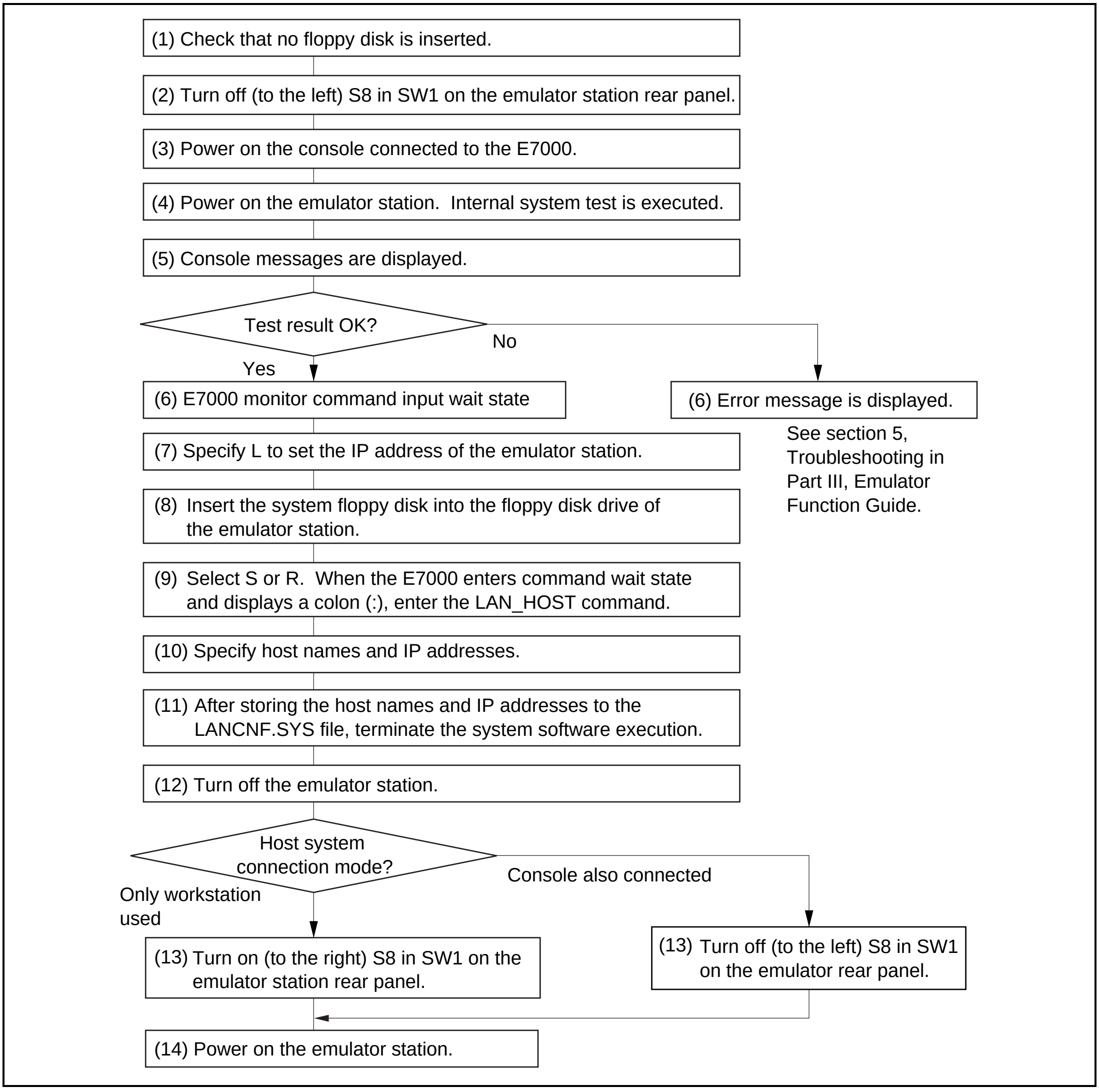

(1) Check that no floppy disk is inserted.
(2) Turn off (to the left) S8 in SW1 on the emulator station rear panel.
(3) Power on the console connected to the E7000.
(4) Power on the emulator station. Internal system test is executed.
(5) Console messages are displayed.
Test result OK?
No
Yes
(6) E7000 monitor command input wait state
(6) Error message is displayed.
See section 5, Troubleshooting in Part III, Emulator Function Guide.
(7) Specify L to set the IP address of the emulator station.
(8) Insert the system floppy disk into the floppy disk drive of the emulator station.
(9) Select S or R. When the E7000 enters command wait state and displays a colon (:), enter the LAN_HOST command.
(10) Specify host names and IP addresses.
(11) After storing the host names and IP addresses to the LANCNF.SYS file, terminate the system software execution.
(12) Turn off the emulator station.
Host system connection mode?
Console also connected
Only workstation used
(13) Turn on (to the right) S8 in SW1 on the emulator station rear panel.
(13) Turn off (to the left) S8 in SW1 on the emulator rear panel.
(14) Power on the emulator station.

Figure 3.15 Power-On Procedure for LAN Interface

The following describes the power-on procedure shown in figure 3.15.

Steps (1) to (7):

The optional LAN board supports the TCP/IP protocol. When the host system is connected to the E7000 with the LAN interface, the IP address (internet address) of the E7000 must be specified. To specify the address, turn off console interface switch S8 in switch set SW1 on the emulator station rear panel, and connect the E7000 to a console with the console interface cable supplied with the E7000. Check that no floppy disk is in the floppy disk drive. Turn on the power at the emulator station rear panel. The console displays the following messages and the E7000 waits for command input.

```
E7000 MONITOR Vn.m
Copyright (C) 1991 Hitachi, Ltd.
Licensed Material of Hitachi, Ltd.

TESTING
RAM 0123

**  E7000 SYSTEM LOADING  **

***  FD NOT READY  ***

START E7000
  S: START E7000
  R: RELOAD & START E7000
  B: BACKUP FD
  F: FORMAT FD
  L: SET LAN PARAMETER
  T: START DIAGNOSTIC TEST
     (S/R/B/F/L/T) ? _
```

After the above messages are displayed, press L and then the (RET) key. The E7000 prompts IP address input. 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'80000001 (H' represents hexadecimal), the E7000 will display the address as follows:

```
: IP ADDRESS = 128.0.0.1 _
```

Enter the IP address. For example, to specify H'80000002, enter as follows:

```
: IP ADDRESS = 128.0.0.1   128.0.0.2  (RET)
```

After entering the IP address, press the (RET) key. The console will display a message, which shows that IP address specification has been completed.

The host system name and IP address of the E7000 must be specified in the network database for the host system. When the host system uses UNIX*, the host name and IP address should generally be specified in the /etc/hosts file. For details, refer to the host system user’s manual.

Note: Unix is a registered trademark of UNIX System Laboratories, Inc.

Steps (8) to (12):

To transfer data between the host system and the E7000, initiate the FTP server to connect the host system to the E7000. Before the FTP server is initiated, the host name and the IP address of the host system must be stored in the LANCNF.SYS file in the E7000 system disk. The following describes how to specify the host name and IP address.

Insert the E7000 system disk into the floppy disk drive of the emulator station and enter S or R to initiate the system program while messages are displayed on the console. The system disk must be write-enabled. When the E7000 prompts for input with a colon (:), enter the LAN_HOST command.

```
: LAN_HOST; S  (RET)
```

When the LAN_HOST command is entered, the following message is displayed on the console.

```
NO    <HOST NAME>    <IP ADDRESS>     NO    <HOST NAME>    <IP ADDRESS>
 01      xxxxxx          xxxxxx        02      xxxxxx          xxxxxx
 03      xxxxxx          xxxxxx        04      xxxxxx          xxxxxx
 05      xxxxxx          xxxxxx        06      xxxxxx          xxxxxx
 07      xxxxxx          xxxxxx        08      xxxxxx          xxxxxx
 09      xxxxxx          xxxxxx
PLEASE SELECT NO ? _
```

Up to nine pairs of host names and IP addresses can be specified. Input a number from 1 to 9.

```
PLEASE SELECT NO ? 1 (RET)
```

The E7000 prompts for the host name. Enter a name with six characters.

```
01 HOST NAME   xxxxxx   <name of host machine> (RET)
```

After that, the E7000 prompts for the IP address. Enter the IP address in decimal.

```
01 IP ADDRESS   xxxxxx   <IP address of host machine> (RET)
```

After the IP address has been specified, the E7000 will prompt for another selection number. When connecting more than one host system, continue specifying the host names and IP addresses. To terminate input, enter as follows:

```
PLEASE SELECT NO ? . (RET)
```

After (RET) is entered, the E7000 enters overwrite confirmation wait state. To store the host names and IP addresses, enter as follows:

```
OVER WRITE(Y/N) ?   Y (RET)
```

The specified host names and IP addresses are stored in the LANCNF.SYS file in the E7000 system disk. Once they have been stored, remove the system disk from the floppy disk drive, set write protection to the disk, and then turn off the E7000.

Steps (13) and (14):

The host system can be connected to the E7000 in the following two modes.

- Only workstation is used
  The TELNET server is used.

  - Turn on (to the right) SW1-S8 on the emulator station rear panel.
  - Power on the E7000.
  - Execute the TELNET command on the workstation.

  Note: After initiating the E7000, enter as follows to cancel local echo on the workstation:

  ```
  CTRL +]
  telnet>mode character (RET)
  ```

  Some workstations sometimes do not accept CTRL+S or CTRL+Q key input. In this case, when the TELNET supports a toggle localflow function, specify it.

- Console connected in addition to workstation
  The console is connected to the RS-232C connector on the emulator station.

  - Turn off (to the left) SW1-S8 on the emulator station rear panel.
  - Power on the console.
  - Power on the E7000.

  After the system program is initiated, the FTP server can be initiated from the console.

For operations after power-on, refer to section 3.5, E7000 Monitor Initiation, section 3.6, Floppy Disk Backup, and section 3.7, E7000 System Program Initiation.

3.4.2 Power-On Procedure for RS-232C Interface

Figure 3.16 shows the power-on procedure when the RS-232C interface is used.

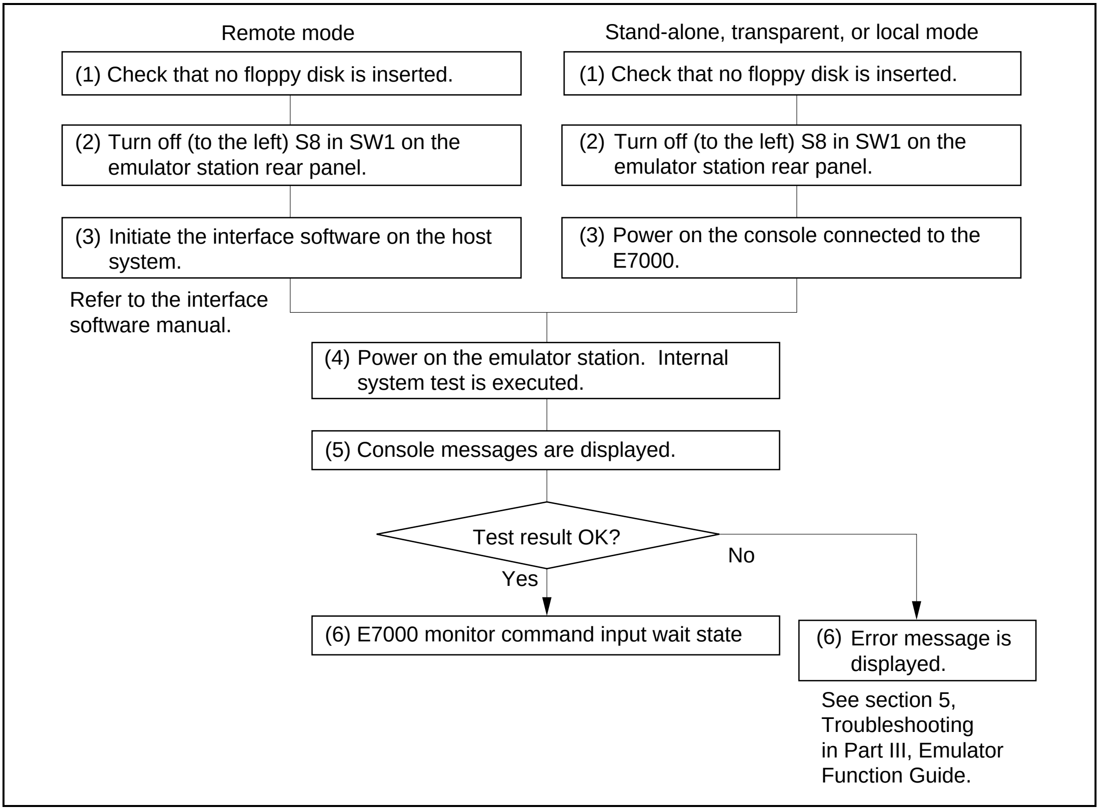

Remote mode
Stand-alone, transparent, or local mode
(1) Check that no floppy disk is inserted.
(1) Check that no floppy disk is inserted.
(2) Turn off (to the left) S8 in SW1 on the emulator station rear panel.
(2) Turn off (to the left) S8 in SW1 on the emulator station rear panel.
(3) Initiate the interface software on the host system.
Refer to the interface software manual.
(3) Power on the console connected to the E7000.
(4) Power on the emulator station. Internal system test is executed.
(5) Console messages are displayed.
Test result OK?
Yes
No
(6) E7000 monitor command input wait state
(6) Error message is displayed.
See section 5, Troubleshooting in Part III, Emulator Function Guide.

Figure 3.16 Power-On Procedure for RS-232C Interface

For operations after power-on, refer to section 3.5, E7000 Monitor Commands, section 3.6, Floppy Disk Backup, and section 3.7, E7000 System Program Initiation.

3.5 E7000 Monitor Commands

3.5.1 E7000 Monitor Initiation

The E7000 supports the six monitor commands listed in table 3.7. These commands load the E7000 system program or diagnostic program, format or back up floppy disks, and set an IP address. When the E7000 is turned on, it displays the following message and waits for the monitor command input.

Enter a command to be executed (table 3.7). If the system disk is inserted when the E7000 is turned on, the E7000 automatically loads the system program without entering the command input wait state.

When the E7000 is turned on without the system floppy disk, the following messages are displayed.

Console Messages:

```
E7000 MONITOR Vn.m
Copyright (C) 1991 Hitachi, Ltd.                (a)
Licensed Material of Hitachi, Ltd.

TESTING                                         (b)
RAM 0123

**  E7000 SYSTEM LOADING  **
                                                (c)
***  FD NOT READY  ***

START E7000
  S: START E7000
  R: RELOAD & START E7000
  B: BACKUP FD
  F: FORMAT FD                                  (d)
  L: SET LAN PARAMETER
  T: START DIAGNOSTIC TEST
    (S/R/B/F/L/T) ? _
```

Descriptions:

(a) E7000 monitor start message. Vn.m is the E7000 monitor's version number. If this message is not displayed, determine what is wrong by reading section 5, Troubleshooting in Part III, Emulator Function Guide.

(b) The E7000 internal system is being tested. A number from 0 to 3 is displayed when each of the four MCU internal RAM blocks has been tested. If an error occurs, the following messages are displayed:

```
***  RAM ERROR ADDR = xxxxxxxx  W-DATA = xxxxxxxx  R-DATA = xxxxxxxx
***  xxxxx REGISTER ERROR  W-DATA = xxxx  R-DATA = xxxx
```

If these messages are displayed, refer to section 5, Troubleshooting in Part III, Emulator Function Guide.

(c) E7000 system program load message. Because the system floppy disk is not inserted, *** FD NOT READY *** is displayed.

(d) List of E7000 monitor commands. Enter the required command at the cursor position. These commands are described in table 3.7. When a B, F, L, or T command is specified, the E7000 will prompt for another command after execution is completed. After the system program has been loaded by an S or R command, the QUIT command ends the system program execution and returns the E7000 monitor to command input wait state.

Table 3.7 E7000 Monitor Commands

Command	Function	Reference Section
S	E7000 system program initiation Initiates the system program. When the system program has not been loaded, loads it from the floppy disk and then initiates the system.	Section 3.7, E7000 System Program Initiation
R	E7000 system program reload Loads and initiates a different system program from the loaded system program.	Section 3.7, E7000 System Program Initiation
B	Floppy disk backup and verification Backs up or verifies a floppy disk.	Section 3.6.2, Floppy Disk Backup and Verification
F	Floppy disk format Formats a floppy disk.	Section 3.6.1, Floppy Disk Formatting
L	IP address specification Specifies the IP address.	Section 3.4.1, Power-On Procedure for LAN Interface
T	Diagnostic Program Initiation Loads and initiates the diagnostic program in the E7000 system floppy disk. If a problem occurs, use this command to initiate the diagnostic program.	Attached diagnostic program manual

Each monitor command is described in the following pages. The input format for monitor commands is generally as follows:

<command name> (RET)

(RET): (RET) key input

3.5.2	**S**	**Initiates the E7000 system program**

Command Format

- Initiation : S (RET)

Description

- Initiation

  Loads the E7000 system program from the system disk. If the system program has already been loaded, it initiates the E7000 system program. Use this command to re-initiate the E7000 system program after modifying the operating environment with the LAN_HOST or MODE command. Also use this command to load and initiate the E7000 system program in normal cases, with the following exceptions.

  — An illegal system program is loaded by mistake.
  — A system program is reloaded due to system error.

  Refer to section 3.5.3, R, for details on system program initiation for the above two exceptions.

Example

To initiate the E7000 system program:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? S (RET)

** E7000 SYSTEM LOADING **
```

3.5.3	**R**	**Reloads and initiates the E7000 system program**

Command Format

- Initiation : R (RET)

Description

- Initiation

  Loads the E7000 system program from the system disk and initiates it even if the system program has already been initiated. Use this command to reload the E7000 system program due to system program error or to initiate another E7000 system program whose parameters are different from those in the program that has been loaded.

Example

To reload and initiate the E7000 system program:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? R (RET)

** E7000 SYSTEM LOADING **
```

3.5.4	**B**	**Backs up and verifies the E7000 system program**

Command Format

- Backing up : B (RET)
- Verification : B; V (RET)

Description

- Backing up

Backs up the E7000 system disk. Use the E7000 system disk backed up with this command. First, format the backup disk with the F command (section 3.5.5) before backing it up with this command.

Since the E7000 has only one floppy disk drive, the user must back up the floppy disk, alternately exchanging the source disk with the target disk.

Note that the original floppy disk is called the source floppy disk and the disk to be backed up or verified is called the target floppy disk.

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

*** FD BACKUP ***
SET SOURCE FD (Y/N) (a) (RET)
SET TARGET FD (Y/N) (b) (RET)
SET NEXT TARGET FD (Y/N) (c) (RET)
```

(a) The source floppy insertion confirmation message. Insert the source floppy disk, and enter Y and the (RET) key to read data from the source floppy disk to the E7000 memory. Enter N to terminate this command.

(b) The target floppy disk insertion message displayed after read. Exchange the target floppy disk with the source floppy disk, and enter Y and the (RET) key to write data from the E7000 memory to the target floppy disk. Enter N to terminate this command.

(c) Another target floppy disk insertion confirmation message displayed after backup. Insert another target floppy disk, and enter Y and the (RET) key to write data from the E7000 memory to another target floppy disk. Enter N to terminate this command.

- Verification

  Verifies the floppy disk in a manner similar to floppy disk back-up. If a verification error occurs, the following error message is displayed.

```
<SECTOR>  <OFFSET>  <SOURCE>  <TARGET>
  0004       045       FF'.'     42'B'
   (a)       (b)        (c)       (d)
```

(a) Serial number of sector containing a verification error

(b) Offset from the sector containing a verification error

(c) Source floppy disk data (hexadecimal and ASCII)

(d) Target floppy disk data (hexadecimal and ASCII)

Examples

1. To back up the E7000 system program:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? B (RET)

*** FD BACKUP ***
SET SOURCE FD (Y/N) Y (RET)
SET TARGET FD (Y/N) Y (RET)
SET NEXT TARGET FD (Y/N) N (RET)
*** BACKUP END ***

START E7000
 S: START E7000
 R: RELOAD & START E7000
 B: BACKUP FD
 F: FORMAT FD
 L: SET LAN PARAMETER
 T: START DIAGNOSTIC TEST
    (S/R/B/F/L/T)?
```

B

2. To verify the E7000 system program:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? B;V (RET)

*** FD VERIFY ***
SET SOURCE FD (Y/N) Y (RET)
SET TARGET FD (Y/N) Y (RET)
SET NEXT TARGET FD (Y/N) N (RET)
*** VERIFY END ***

START E7000
 S: START E7000
 R: RELOAD & START E7000
 B: BACKUP FD
 F: FORMAT FD
 L: SET LAN PARAMETER
 T: START DIAGNOSTIC TEST
    (S/R/B/F/L/T)?
```

3.5.5	**F**	**Formats the floppy disk**

Command Format

- Formatting : F (RET)

Description

- Formatting

  Formats the floppy disk. To back up the system disk, use a disk formatted with this command. Set the floppy disk to be formatted and enter this command. The following messages are displayed to confirm the volume label name and format.

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

   VOLUME LABEL : <volume name>    (a)
   START FORMAT (Y/N) (b) (RET)
```

(a) <volume name> is displayed. A space is displayed if the floppy disk has no volume label name.

(b) Format confirmation message is displayed.
  Y: Starts formatting
  N: Terminates this command

Example

To format a floppy disk:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? F (RET)

 VOLUME LABEL : WORK
 START FORMAT (Y/N) Y (RET)
```

3.5.6 L — Sets an E7000 IP address

Command Format

- Setting : L (RET)

Description

- Setting

Sets an E7000 IP address. This is required to connect the E7000 to the host system through the optional LAN board. For details, refer to section 3.4.1, Power-on Procedure for LAN Interface, and the manual provided with the LAN board. If the L command is entered, the E7000 displays the current IP address and waits for a new IP address input. Note that the IP address must be entered in decimal.

```
      (S/R/B/F/L/T) ? L (RET)
:IP ADDRESS = xxx.xxx.xxx.xxx (a) (RET)
```

(a) A new IP address in decimal. To not change the IP address, enter only the (RET) key.

Example: 128.1.1.101

Example

To set an E7000 IP address:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? L (RET)

 :IP ADDRESS = 0.0.0.0 128.1.1.9 (RET)

START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ?
```

3.5.7 T Initiates the diagnostic program

Command Format

- Initiation : T (RET)

Description

- Initiation

  Loads the diagnostic program from the system disk and initiates it.

Example

To initiate the diagnostic program:

```
START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
   (S/R/B/F/L/T) ? T (RET)

** E7000 TM LOADING **
```

3.6 Floppy Disk Backup

Before using the E7000 system floppy disk, prepare a backup in case the original is damaged. This section describes the disk formatting and backup procedure.

3.6.1 Floppy Disk Formatting

Format a backup disk for the E7000 system according to the messages displayed on the console, using the procedure shown below. Only use 2HD (double sided, high density, double track) floppy disks. Do not remove the disk while the disk drive is operating.

Procedure	Console Messages
1. E7000 monitor command prompt.	START E7000 S: START E7000 R: RELOAD & START E7000 B: BACKUP FD F: FORMAT FD L: SET LAN PARAMETER T: START DIAGNOSTIC TEST (S/R/B/F/L/T) ? _
2. Insert the floppy disk to be formatted.	
3. Enter F and (RET).	(S/R/B/F/L/T) ? F (RET)
4. A volume label is displayed if there is a volume on the floppy disk.	VOLUME LABEL : xx xx
5. Format start confirmation message. Enter Y (RET) to format, otherwise enter N (RET).	START FORMAT (Y/N) ? Y (RET)
6. The E7000 monitor prompts for another command after formatting is completed.	

If an error occurs during formatting, refer to section 12, Error Messages, for details.

3.6.2 Floppy Disk Backup and Verification

Use the following procedures to back up and verify a floppy disk.

Backup:

Procedure	**Console Messages**
1. E7000 monitor command prompt.	START E7000 S: START E7000 R: RELOAD & START E7000 B: BACKUP FD F: FORMAT FD L: SET LAN PARAMETER T: START DIAGNOSTIC TEST (S/R/B/F/L/T) ? _
2. Enter backup command B (RET).	(S/R/B/F/L/T) ? B (RET)
3. Backup start message and source floppy disk insertion request message.	*** FD BACKUP *** SET SOURCE FD (Y/N) ? _
4. Insert the source floppy disk.	
5. Enter Y (RET). Data is read from the source floppy disk into the E7000 internal RAM.	SET SOURCE FD (Y/N) ? Y (RET)
6. When read is completed, the target floppy disk insertion message is displayed. Exchange the floppy disk.	SET TARGET FD (Y/N) ? _
7. Enter Y (RET). The E7000 internal RAM contents are written to the target floppy disk.	SET TARGET FD (Y/N) ? Y (RET)
8. When write is completed, the E7000 asks whether to back up another target floppy disk or complete backup operation.	SET NEXT TARGET FD (Y/N) ? _
9. To make a backup on another target floppy disk, insert the new target disk and then enter Y (RET). Repeat steps 6 to 9.	SET NEXT TARGET FD (Y/N) ? Y (RET)
10. Enter N (RET) to complete backup operation.	SET NEXT TARGET FD (Y/N) ? N (RET)
11. Backup completion message. The E7000 monitor prompts for another command.	*** BACKUP END ***

Verification:

Procedure — **Console Messages**

1. E7000 monitor command prompt.

```
START E7000
  S: START E7000
  R: RELOAD & START E7000
  B: BACKUP FD
  F: FORMAT FD
  L: SET LAN PARAMETER
  T: START DIAGNOSTIC TEST
      (S/R/B/F/L/T) ? _
```

2. Enter verification command B;V (RET).

```
      (S/R/B/F/L/T) ? B;V (RET)
```

3. Verification start message and source floppy disk insertion request message.

```
***  FD VERIFY  ***
SET SOURCE FD  (Y/N) ? _
```

4. Insert the source floppy disk.

5. Enter Y (RET). Data is read from the source floppy disk into the E7000 internal RAM.

```
SET SOURCE FD (Y/N) ? Y (RET)
```

6. When read is completed, the target floppy disk insertion message is displayed. Exchange the floppy disk.

```
SET TARGET FD  (Y/N) ? _
```

7. Enter Y (RET). The contents of E7000 internal RAM are compared with the contents of the target floppy disk.

```
SET TARGET FD (Y/N) ? Y (RET)
```

8. When comparison is completed, the E7000 asks whether to verify another target floppy disk or complete verification operation.

```
SET NEXT TARGET FD  (Y/N) ? _
```

9. To verify another target floppy disk, insert the new target disk and then enter Y (RET).Repeat steps 6 to 9.

```
SET NEXT TARGET FD (Y/N) ? Y (RET)
```

10. Enter N (RET) to complete verification operation.

```
SET NEXT TARGET FD (Y/N) ? N (RET)
```

11. Verification completion message. The E7000 monitor prompts for another command.

```
***  VERIFY END  ***
```

Notes: 1. Any differences in the contents of the disks are displayed in the following format:

```
<SECTOR>     <OFFSET>     <SOURCE>     <TARGET>
  xxxx          xxx         xx 'x'       xx 'x'
  (a)           (b)        (c) (d)      (e) (f)
```

(a) Serial number of sector containing the difference, beginning at 0 (hexadecimal)

(b) Offset in the sector containing the difference (hexadecimal)

(c) Source floppy disk contents (hexadecimal)

(d) Source floppy disk contents in ASCII characters

(e) Target floppy disk contents (hexadecimal)

(f) Target floppy disk contents in ASCII characters

2. During both backup and verification, entering N (RET) in response to the floppy disk insertion request message terminates command execution and returns the E7000 to command input wait state.

3. If a floppy disk error occurs during backup or verification, an error message is displayed and command execution is aborted. Refer to section 12, Error Messages, for details.

3.7 E7000 System Program Initiation

When the E7000 system floppy disk is not inserted after the E7000 is turned on, the E7000 monitor enters command input wait state and the E7000 system program must be loaded and initiated by monitor commands. If the system floppy disk is inserted immediately after power-on, the system program is automatically loaded and initiated.

3.7.1 Initiation on E7000 Monitor

If S or R is entered, followed by (RET), when the E7000 is in monitor command input wait state, the E7000 system program is loaded from the system disk and initiated.

Table 3.8 E7000 System Program Initiation Commands

Command	Description
S	Loads and initiates the system program from the E7000 system disk. If the E7000 system program is already loaded, the system program is initiated immediately.*
R	Reloads and initiates the E7000 system program.

Note: * This situation occurs when the system program is initiated and then terminated with the QUIT command. However, if the E7000 monitor F (format), B (backup or verification), or T (diagnostic test) command has been executed, or when the system program has been forcibly terminated by a clock error, the system program is reloaded.

Display at E7000 System Program Initiation

```
START E7000
  S: START E7000
  R: RELOAD & START E7000
  B: BACKUP FD
  F: FORMAT FD                                  (a)
  L: SET LAN PARAMETER
  T: START DIAGNOSTIC TEST
       (S/R/B/F/L/T) ? { S } (RET)
                       { R }

**  E7000 SYSTEM LOADING **                     (b)

SH7604 E7000 (HSxxxx EPDxxSF) Vn.m
Copyright (C) Hitachi, Ltd. 1993                (c)
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING                      (d)
LAN IP ADDRESS FILE LOADING                     (e)
```

```
MODE CHECK                                              (f)
HARD WARE REGISTER READ/WRITE CHECK                     (g)
POD SYSTEM LOADING                                      (h)
EMULATOR POD TEST                                       (i)
**  RESET IN BY E7000  !                                (j)
CLOCK = xxxx                                            (k)
MASTER  MODE = xx (MD 5-0 = xx)                       ┐
MODE  SET = x                                         ┘ (l)
REMAINS EMULATION MEMORY  S = D'xxxxxkB                 (m)

WARM OR COLD START                                    ┐
  file name          : WARM START                     │
  return             : COLD START                     │ (n)
  (file name/return) ? { <file name> (RET) }          │
                       { (RET)             }          ┘

:                                                       (o)
```

Description

(a) E7000 command input request message. Insert the E7000 system disk and enter S. Enter R if loading another E7000 system program.

(b) The E7000 system program is being loaded from the floppy disk.

(c) Start message of the E7000 system program. Vn.m is the version number.

(d) Configuration file is being loaded.

(e) IP address file for the LAN is being loaded.

(f) The specified E7000 mode is compared with the mode selection pin status on the user system. If the E7000 is in master mode and the mode selection pin status is in slave mode, the SH-2 operating mode cannot be set to the mode specified in the configuration file. In this case, initiation will be aborted and the following message will be displayed:

```
USER MODE = SLAVE
```

Either change the E7000 mode to slave mode with the MODE command or change the mode selection pin status to master mode.

(g) Emulator station hardware test start message. If there is an error in the emulator station, an error message is displayed.

(h) The program to be executed in the emulator pod is being loaded from the floppy disk.

(i) Emulator pod test start message. If there is an error in the emulator pod, an error message is displayed.

(j) A RES signal has been input to the SH-2.

(k) Specified clock. If the user system is ready, the user system's clock is used. If the user system's clock is not ready but the 6.144-MHz E7000 internal clock is ready, the E7000 internal clock is used.

(l) SH-2 operating mode and user system mode selection pin state. The operating mode, the CS0 area bus width, and master/slave are those previously set with the MODE command (saved in configuration file). For details, refer to section 7.2.5, MODE.

(m) Remaining emulation memory size

(n) Specify either WARM START*[1] or COLD START*[2] as follows:

WARM START: Specify the file name containing recovery information.
COLD START: Press the (RET) key.

(o) E7000 system program prompt. An E7000 system program command can now be entered.

Notes: 1. WARM START recovers the information saved in a file when the E7000 system program was terminated by a QUIT command. (For details, refer to section 7.2.29 QUIT.) The recovery information is listed below.

- Software breakpoints
- Hardware break conditions, and trace stop and acquisition conditions
- Memory map information
- Configuration information
- Symbol information

2. COLD START initializes the above emulation information.

3.7.2 Automatic Initiation of E7000 System Program

If the E7000 system disk is inserted after E7000 internal system test at power-on has been completed, the E7000 system program is automatically loaded.

Console display **Procedure**

```
E7000 MONITOR Vn.m
Copyright (C) 1991 Hitachi, Ltd.
Licensed Material of Hitachi, Ltd.

TESTING
RAM 0123
```

After E7000 power-on, insert the system disk when this message is displayed.

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

E7000 system program is being loaded.

```
SH7604 E7000  (HSxxxx EPDxxSF)  Vn.m
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
LAN IP ADDRESS FILE LOADING
MODE CHECK
HARD WARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TEST
**  RESET IN BY E7000 !
CLOCK = xxxx
MASTERMODE = xx (MD 5-0 = xx)
MODE  SET = x
REMAINS EMULATION MEMORY   S = D'xxxxxkB

WARM OR COLD START
  file name           : WARM START
  return              : COLD START
  (file name/return) ? { <file name> (RET) }
                       { ((RET)           }
:
```

System program is initiated. (Refer to section 3.5, E7000 Monitor Commands.)

Section 4 Operating Examples

Section 4.1, Basic Examples, and section 4.2, Application Examples, include explanations based on the following user program.

```
ADDR        CODE     LABEL     MNEMONIC    OPERAND
01001000    E00A               MOV         #0A,R0
01001002    E101               MOV         #01,R1
01001004    E201               MOV         #01,R2
01001006    D405               MOV.L       0100101C,R4
01001008    6323     !LOOP     MOV         R2,R3
0100100A    321C               ADD         R1,R2
0100100C    2426               MOV.L       R2,@-R4
0100100E    6133               MOV         R3,R1
01001010    70FF               ADD         #FF,R0
01001012    8800               CMP/EQ      #00,R0
01001014    8BF8               BF          !LOOP
01001016    0009               NOP
01001018    AFFE     !LABEL    BRA         !LABEL
0100101A    0009               NOP
0100101C    0F10               .DATA.W     0F10
0100101E    0000               .DATA.W     0000
```

These examples assume that the emulator station is connected to the LAN host system with the Telnet and that the user program is downloaded from the host system to the E7000. Therefore, store the program in the host system before initiating the E7000. In these examples, the host name is HITACHI, and the IP address is 128.0.0.1.

Initiate the E7000 by the following procedure:

Operations	Console Message
1. Insert the E7000 system disk into the floppy disk drive of the emulator station, and turn on the power.	E7000 MONITOR Vn.m Copyright (C) 1991 Hitachi, Ltd. Licensed Material of Hitachi, Ltd. TESTING RAM 0123

2. The console displays the message shown on the right when the E7000 starts operation normally.
   (If the console does not display this message, take corrective action as described in section 5, Troubleshooting.)

```
** E7000  SYSTEM LOADING **


SH7604 E7000 (HSxxxxEPDxxSF) Vm. n
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
LAN IP ADDRESS FILE LOADING
MODECHECK
HARD WARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TESTING
** RESET IN BY E7000 !
CLOCK = xxxx
MASTER MODE = xx (MD 5-0= xx)
MODE SET = x
REMAINS EMULATIONMEMORYS=D' xxxxxkB

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

3. Enter (RET).

4.1 Basic Examples

4.1.1 Preparing for Connection of LAN Host System

Before connecting the host system, specify the host system name and the IP address by the following procedure:

Operations **Console Message**

1. Specify the IP address of the host system to which the E7000 is to be connected by the FTP command. Enter LAN_HOST;S (RET), and the console will display the host system names and IP addresses already specified and wait for the user to enter a selection number.

```
:LAN_HOST;S (RET)
```

```
:LAN_HOST;S (RET)
NO    <HOST NAME>    <IP ADDRESS>    NO    <HOST NAME>    <IP ADDRESS>
01       HOST_A         128.1.1.0    02       HOST_B         128.1.1.1
03       HOST_C         128.1.1.2    04       HOST_D         128.1.1.3
05       HOST_E         128.1.1.4    06       HOST_F         128.1.1.5
07       HOST_G         128.1.1.6    08       HOST_H         128.1.1.7
09       HOST_I         128.1.1.8
PLEASE SELECT NO? _
```

2. Enter 1 (RET) as the selection number, HITACHI (RET)as the host system name, and 128.0.0.1 (RET) as the IP address. After that, the console prompts the user to select another number. Enter . (RET) to exit interactive input mode. After the host system name and IP address have been specified, the console asks if the specified name and address should be overwritten to the LANCNF.SYS file in the system disk. To store them, enter Y (RET).

```
PLEASE SELECT NO? 1 (RET)
01   HOST NAME  HOST_A   HITACHI  (RET)
01   IP ADDRESS   128.1.1.0   128.0.0.1  (RET)
PLEASE SELECT NO? . (RET)
OVERWRITE (Y/N)?  Y (RET)
```

3. After the name and address have been stored in the LANCNF.SYS file, the E7000 system program automatically terminates. Therefore, restart the E7000 system.

4. Enter S (RET) to re-initiate the system.

```
START E7000
  S : START E7000
  R : RELOAD & START E7000
  B : BACKUPFD
  F : FORMATFD
  L : SET LAN PARAMETER
  T : START DIAGNOSTIC TEST
          (S/R/B/F/L/T) ?  S (RET)
```

5. Enter (RET).

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

Note: The above host name and IP address are examples. Specify the actual host system name and IP address according to the system.

4.1.2 Specifying the SH-2 Operating Mode

Specify the E7000 operating mode and the SH-2 operating mode by the following procedure:

Operations / **Console Message**

1. Enter MODE;C (RET) to specify the E7000 operating mode.

```
:MODE;C (RET)
```

2. The console displays the message shown on the right. To select mode 8 of the SH-2 and use the configuration file data to set the SH-2 operating mode, for example, enter 8 (RET) and C (RET).

```
E7000MODE  (MD5-0) = xx ? _
MODE SET (C:CONFIGURATION/U:USER/
                    M:MASTER-SLAVE) = M ? _

E7000MODE  (MD5-0) = xx ? 8 (RET)
MODE SET (C:CONFIGURATION/U:USER/
                    M:MASTER-SLAVE) = M ? C (RET)
```

3. After the above specification has been completed, the console asks if the mode settings should be stored in the configuration file. To store the mode settings, enter Y (RET). After that, the E7000 operates in the mode specified above whenever initiated with this system disk. To correct a mis-typed mode number, return to step 1 above before entering Y (RET) and repeat the procedure. Remove the write protect from the system floppy disk before storing the mode settings in the configuration file.

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

CONFIGURATION WRITE OK ? (Y/N) ? Y (RET)
```

4. After the mode settings have been stored in the configuration file, the E7000 system program automatically terminates.

```
START E7000
  S : START E7000
  R : RELOAD & START E7000
  B : BACKUP FD
  F : FORMAT FD
  L : SET LAN PARAMETER
  T : START DIAGNOSTIC TEST
        (S/R/B/F/L/T) ?  _
```

5. Enter S (RET) to re-initiate the system program.

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

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

6. Enter (RET).

```
  (file name/return) ? (RET)
```

4.1.3 Allocating Standard Emulation Memory and Specifying Attributes

In order to load the user program to memory, allocate the standard emulation memory in the pods by the following procedure:

Operations | **Console Message**

Enter MAP 1000000 101FFFF;S (RET) to allocate the standard emulation memory to addresses H'1000000 to H'101FFFF. The console displays the message shown on the right, which indicates that the memory allocation has been completed.

```
:MAP 1000000 101FFFF;S (RET)

 REMAINS EMULATION MEMORY     S=D' 0384kB
```

Enter MAP (RET) and the console displays the attributes of all the memory areas.

```
:MAP (RET)
 01000000 - 0101FFFF;S    21000000 - 2101FFFF;S
 INTERNAL I/O    =        E0000000 - FFFFFFFF
 REMAINS EMULATION MEMORY     S=D' 0384kB
```

4.1.4 Loading the User Program

Load the user program from the host system to the E7000 by the following procedure:

Operations	Console Message
1. Enter FTP <host name> (RET) to connect the E7000 and the host system with the FTP server.	:FTP <host name> (RET)
2. The console asks for the user name. Enter <user name> (RET).	Username : _ Username : <user name> (RET)
3. The console asks for the password. Enter <password> (RET).	Password : _ Password : <password> (RET)
4. The console displays the message shown on the right, which indicates that the E7000 and the host system have been connected.	login command success FTP>_
5. To load the program, enter LAN_LOAD;S:<file name> (RET). This example assumes that the load module is S type. While loading, the console displays the address to which the program is being loaded as shown on the right.	FTP>LAN_LOAD;S:<file name> (RET) LOADING ADDRESS xxxxxx
6. When the program has been loaded, the console displays the start address of the program (TOP ADDRESS), and its end address (END ADDRESS).	TOP ADDRESS = xxxxxxxx END ADDRESS = xxxxxxxx FTP>_
7. Entering BYE (RET) terminates the FTP server connection. The console will display the message shown on the right.	FTP>BYE (RET) bye command success

Note: The following operations can be performed even when the BYE command is not executed and the FTP prompt is displayed. In this case, be sure not to power down the E7000 before executing the BYE command.

8. The DISASSEMBLE command displays the loaded program. Enter DISASSEMBLE 1001000 100101F (RET).

```
:DISASSEMBLE 1001000 100101F (RET)
 ADDR        CODE      LABEL     MNEMONIC    OPERAND
 01001000    E00A                MOV         #0A,R0
 01001002    E101                MOV         #01,R1
 01001004    E201                MOV         #01,R2
 01001006    D405                MOV.L       0100101C,R4
 01001008    6323      !LOOP     MOV         R2,R3
 0100100A    321C                ADD         R1,R2
 0100100C    2426                MOV.L       R2,@-R4
 0100100E    6133                MOV         R3,R1
 01001010    70FF                ADD         #FF,R0
 01001012    8800                CMP/EQ      #00,R0
 01001014    8BF8                BF          !LOOP
 01001016    0009                NOP
 01001018    AFFE      !LABEL    BRA         !LABEL
 0100101A    0009                NOP
 0100101C    0F10                .DATA.W     0F10
 0100101E    0000                .DATA.W     0000
```

4.1.5 Executing Program

Execute the loaded program by the following procedure:

Operations **Console Message**

1. Enter .SP (RET) then FFFFFFC (RET) as the SP value to set the stack pointer (SP register) to H'FFFFFFC.
   The console then asks for the program counter value. Enter 1001000 (RET) as the program counter value. The console then asks for the status register value. In this example, other registers need not be set or changed, therefore, enter . (RET) to exit this interactive mode.

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

 R15(SP) = xxxxxxxx ? FFFFFFC (RET)
 PC      = xxxxxxxx ? _

 PC      = xxxxxxxx ? 1001000 (RET)
 SR     = xxxxxxxx:- - IIII - - - - ? _

 SR     = xxxxxxxx:- - IIII - - - - ? . (RET)
:_
```

Note: In interactive mode, entering only (RET) makes no change to the currently displayed item, and the next item is displayed. In the above example, entering only (RET) can complete the register modification procedure. The register value can also be directly input without using the interactive mode. For example, to set the stack pointer value directly, enter .SP FFFFFFC (RET).

2. Enter GO (RET) to execute the program from the address pointed by the PC. While the program is executed, the console displays the current program counter value (shown as xxxxxxxx on the right).

```
:GO (RET)
 ** PC = xxxxxxxx
```

3. Enter (BREAK) key or (CTRL) + C keys to terminate program execution. The console displays the contents of the program counter, the status register, the control register, and the general registers R0 to R15 at termination. RUN - TIME shows the duration of program execution from the GO command execution to (BREAK) or (CTRL) + C key input. BREAK KEY shows that the execution has been terminated because (BREAK) or (CTRL) + C was entered.

```
:(BREAK)
 PC = 01001018  SR = 000000F1:- - IIII - - - T
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000000  00000059  00000090  00000059
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’0000H:00M:01S:012345US
 +++:BREAK KEY
:_
```

4.1.6 Software Break

Program execution can be stopped at a particular address by setting a breakpoint as follows:

Operations **Console Message**

1. Enter BREAK 1001010 (RET) to terminate program execution immediately before the instruction at address H'1001010 in the program is executed.

```
:BREAK 1001010 (RET)
```

2. Restart program execution from address H'1001000. This can be done in two ways: one is to enter the start address directly, and the other is to first set the program counter to H'1001000, then enter GO, as described in section 4.1.5, Executing Program.

```
:GO 1001000 (RET)
 ** PC = xxxxxxxx
```

3. The program execution terminates immediately before the instruction at address H'1001010 is executed. The console displays the data shown on the right. The BREAK POINT shows that the program execution terminated because of a software breakpoint.

```
 PC = 01001010  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000009  00000001  00000002  00000001
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’0000H:00M:00S:000006US
 +++:BREAK POINT
:_
```

4.1.7 Single-Step Execution

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

Operations | **Console Message**

1. The program counter points to the next address to be executed when the program execution terminates in the example of section 4.1.6, Software Break. Here, entering STEP (RET) executes only one instruction, and the console displays the information as shown on the right. 01001012 CMP/EQ #00,R0 shows the executed address and mnemonic code, and STEP NORMAL END shows that the single-step execution has terminated.

```
:STEP (RET)




 PC = 01001014  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000009  00000001  00000002  00000001
 R8 - 15 00000000  00000000  00000000  00000000
 01001012                    CMP/EQ     #00, R0
 +++:STEP NORMAL END
:_
```

2. To repeat single-step execution, enter only (RET). This can be repeated until another command is executed.

```
:(RET)
 PC = 01001008  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000000  00000001  00000002  00000001
 R8 - 15 00000000  00000000  00000000  00000000
 01001014                       BF          !LOOP
 +++:STEP NORMAL END
:_
```

4.1.8 Setting Hardware Break Conditions

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

Operations **Console Message**

1. Enter BREAK - (RET) to cancel the breakpoint set in the example in section 4.1.6, Software Break.

```
: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'F0FFFF8, enter BREAK_CONDITION1 A = F0FFFF8 W (RET).

```
:BREAK_CONDITION1 A = F0FFFF8 W (RET)
```

4. Enter GO 1001000 (RET) to start executing the program from address H'1001000. When the break condition is satisfied, the console displays the information shown on the right. BREAK CONDITION1 shows that the program execution has terminated because the break condition was satisfied.

```
:GO 1001000 (RET)
 PC = 01001012  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000008  00000002  00000003  00000002
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’0000H:00M:00S:000010US
 +++:BREAK CONDITION1
:_
```

4.1.9 Displaying Trace Information

Trace information acquired during program execution can be displayed by the following procedure:

Operations **Console Message**

1. Enter TRACE (RET) to see the trace information. The console will display the instruction mnemonic information.

```
:TRACE (RET)
```

```
     IP         ADDR       LABEL     MNEMONIC    OPERAND
  *-D'00076   01001000               MOV         #0A,R0
  *-D'00075   01001002               MOV         #01,R1
  *-D'00074   01001004               MOV         #01,R2
  *-D'00073   01001006               MOV.L       0100101C,R4
  *-D'00072   01001008     !LOOP     MOV         R2,R3
      :           :                   :             :
```

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

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

```
       BP        AB          DB       MA   R/W   ST    IRL   NMI  RES  BRQ  PRB  VCC  CLK
  *           01001000                            MOV              #0A,  R0
   –D'00208   01001000   ******E0   EXT   R    PRG   1111   1    1    1    1    1    04
   –D'00207   01001001   ******0A   EXT   R    PRG   1111   1    1    1    1    1    02
  *           01001002                            MOV              #01,  R1
   –D'00206   01001002   ******E1   EXT   R    PRG   1111   1    1    1    1    1    02
   –D'00205   01001003   ******01   EXT   R    PRG   1111   1    1    1    1    1    02
  *           01001004                            MOV              #01,  R2
   –D'00204   01001004   ******E2   EXT   R    PRG   1111   1    1    1    1    1    02
   –D'00203   01001005   ******01   EXT   R    PRG   1111   1    1    1    1    1    02
                  :                                :                          :
```

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

```
(CTRL)+S
(CTRL)+Q
```

4.2 Application Examples

4.2.1 Break with Pass Count Condition

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

Operations **Console Message**

1. Enter BREAK 1001012 5 (RET) to terminate program execution when address H'1001012 is passed five times.

```
:BREAK  1001012  5 (RET)
```

2. To start execution from address H'1001000, enter GO 1001000 (RET).

```
:GO 1001000 (RET)
```

3. When execution terminates after address H'1001012 is passed five times, the console displays the data shown on the right.

```
 PC = 01001014  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D' 0000H:00M:00S:000027US
 +++:BREAK POINT
:_
```

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

```
:BREAK (RET)
 ADDRESS      CNT       PASS      SYMBOL
 01001012     0005      0005
    (a)        (b)       (c)
```

4.2.2 Conditional Trace

The following procedure can be used to limit the acquisition of trace information during program execution.

Operations **Console Message**

1. To cancel the breakpoint set in the example of section 4.2.1, Break with Pass Count Condition, enter BREAK - (RET).

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

2. Enter TRACE_CONDITION A =1001000:1001014;R (RET) to get trace information only while the program counter is between addresses H'1001000 and H'1001014.

```
:TRACE_CONDITION A= 1001000:1001014;R (RET)
```

3. Enter GO 1001000 (RET) to start executing the program, then (BREAK) key or (CTRL) + C keys to terminate the execution.

```
:GO 1001000 (RET)
 ** PC = xxxxxxxx
 PC = 01001018  SR = xxxxxxxx:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’ 0000H:00M:01S:000027US
 +++:BREAK KEY
:_
```

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

```
:TRACE (RET)

     IP          ADDR       LABEL      MNEMONIC     OPERAND
  *-D'*****     01001000               MOV          #0A,R0
  *-D'*****     01001002               MOV          #01,R1
  *-D'*****     01001004               MOV          #01,R2
```

4.2.3 Parallel Mode

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

Operations **Console Message**

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

```
:GO 1001000 (RET)
 ** PC = xxxxxxxx (RET)
#_
```

2. Enter DUMP 1002000 100200F (RET) to display the memory contents from H'1002000 to H'100200F.

```
#DUMP 1002000 100200F (RET)

<ADDR>                <D   A   T   A>                                <ASCII CODE>
01002000   14 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    “. . . . . . . .”
```

3. Enter MEMORY 1001018 FC (RET) to modify the contents of memory address H'1001018 into FC.

```
#MEMORY 1001018 FC (RET)
```

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

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

5. To terminate program execution, enter (BREAK) key or (CTRL) + C keys.

```
(BREAK)
 PC = 01001018  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’ 0000H:01M:01S:000027US
 +++:BREAK KEY
:_
```

6. Enter DISASSEMBLE 1001000 100101F (RET) to confirm that the program has been changed by memory modification in parallel mode.

```
:DISASSEMBLE 1001000 100101F (RET)
```

```
ADDR        CODE      LABEL     MNEMONIC    OPERAND
01001000    E00A                MOV         #0A,R0
01001002    E101                MOV         #01,R1
01001004    E201                MOV         #01,R2
01001006    D405                MOV.L       0100101C,R4
01001008    6323      !LOOP     MOV         R2,R3
0100100A    321C                ADD         R1,R2
0100100C    2426                MOV.L       R2,@-R4
0100100E    6133                MOV         R3,R1
01001010    70FF                ADD         #FF,R0
01001012    8800                CMP/EQ      #00,R0
01001014    8BF8                BF          !LOOP
01001016    0009                NOP
01001018    AFFC      !LABEL    BRA         1001016
0100101A    0009                NOP
0100101C    0F10                .DATA.W     0F10
0100101E    0000                .DATA.W     0000
```

4.2.4 Searching Trace Information

The TRACE_SEARCH command can be used to search for a particular part of the acquired trace information.

Operations | **Console Message**

Enter TRACE_SEARCH A=1001018 (RET), and the console will only display those parts of the trace information in which the address bus value is H'1001018.

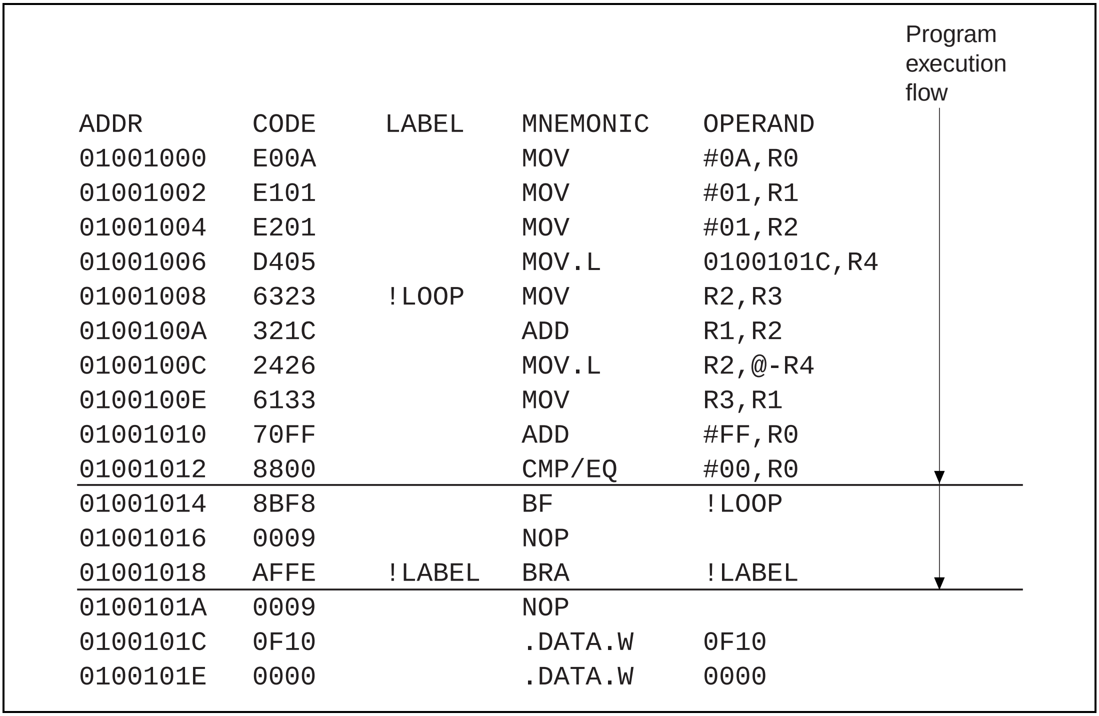
```
:TRACE_SEARCH A=1001018 (RET)
```

```
    BP         AB         DB        MA   R/W  ST   IRL   NMI  RES  BRQ  PRB  VCC  CLK
-D'03392   01001018   00000000   EXT   R   PRG  1111   1    1    1    1    1    02
-D'03879   01001018   00000000   EXT   R   PRG  1111   1    1    1    1    1    02
-D'03766   01001018   00000000   EXT   R   PRG  1111   1    1    1    1    1    02
               :                            :                             :
```

4.2.5 Sequential Software Break

A break can be generated when specified addresses are passed in a specified order, using the BREAK_SEQUENCE command as follows:

Operations	Console Message
1. Enter BREAK_SEQUENCE 1001012 1001018 (RET), which terminates program execution when the instructions at addresses H'1001012 and H'1001018 are executed consecutively in that order, as shown in figure 4.1.	:BREAK_SEQUENCE 1001012 1001018 (RET)



```
ADDR        CODE     LABEL    MNEMONIC    OPERAND
01001000    E00A              MOV         #0A,R0
01001002    E101              MOV         #01,R1
01001004    E201              MOV         #01,R2
01001006    D405              MOV.L       0100101C,R4
01001008    6323     !LOOP    MOV         R2,R3
0100100A    321C              ADD         R1,R2
0100100C    2426              MOV.L       R2,@-R4
0100100E    6133              MOV         R3,R1
01001010    70FF              ADD         #FF,R0
01001012    8800              CMP/EQ      #00,R0
01001014    8BF8              BF          !LOOP
01001016    0009              NOP
01001018    AFFE     !LABEL   BRA         !LABEL
0100101A    0009              NOP
0100101C    0F10              .DATA.W     0F10
0100101E    0000              .DATA.W     0000
```

Figure 4.1 Program Execution Flow

2. Enter GO 1001000 (RET) to execute the program. When the specified condition is satisfied, execution terminates, and the console displays the data shown on the right. The BREAK SEQUENCE shows that execution has terminated because the condition specified in the BREAK_SEQUENCE command has been satisfied.

```
:GO 1001000 (RET)
 PC = 01001018  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D' 0000H:00M:00S:000027US
 +++:BREAK SEQUENCE
:_
```

3. Enter TRACE (RET) to confirm the executed instructions.

```
:TRACE (RET)
```

```
      IP           ADDR      LABEL     MNEMONIC     OPERAND
*-D'00005      0100100E                MOV          R3,R1
*-D'00004      01001010                ADD          #FF,R0
*-D'00003      01001012                CMP/EQ       #00,R0
*-D'00002      01001014                BF           !LOOP
*-D'00001      01001016                NOP
* D'00000      01001018   !LABEL      *   BREAK   *
:
```

Appendix A Floppy Disk Drive Specifications

Table A.1 summarizes the specifications of the 3.5-inch floppy disk drive installed in the E7000.

Table A.1 3.5-Inch Floppy Disk Drive Specifications

Item	Specification
Storage capacity	Approx. 1.2 Mbytes (512 bytes × 15 sectors × 160 tracks) Double-sided, high density, double tracks
Recording method	MFM type
Recording format	IBM format (512 bytes/sector, 15 sectors/track)
Recommended disks	MF2-256HD (Maxell)

Appendix B Connector Specifications

This section describes console connector locations and console connecting methods.

B.1 Console Connector

Figure B.1 shows pin locations in the console's connector. Table B.1 lists signal names and their usages.

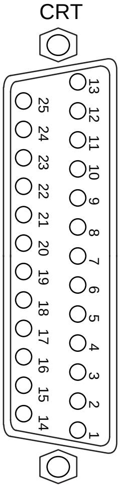


Figure B.1 Console Connector Pin Locations

Table B.1 Signal Names and Usage of Console Connector

Pin No.	Signal Name	Usage
1	Frame Ground (FG)	Connected to the E7000's frame ground
2	Receive Data (RD)	Data receive line
3	Transmit Data (TD)	Data transmit line
4	Clear To Send (CTS)	Not used
5	Request To Send (RTS)	High when E7000's power is on
6	Data Terminal Ready (DTR)	High when E7000's power is on
7	Signal Ground (SG)	Signal ground
8	Data Carrier Detect (DCD)	High when E7000's power is on
9–19	Not connected	—
20	Data Set Ready (DSR)	Not used
21–25	Not connected	—

B.2 Host System Connector

Figure B.2 shows pin locations in the host system's connector. Table B.2 lists signal names and their usages.

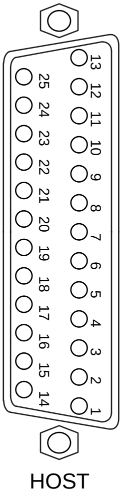


Figure B.2 Host System Connector Pin Locations

Table B.2 Signal Names and Usage of Host System Connector

Pin No.	Signal Name	Usage
1	Frame Ground (FG)	Connected to E7000 frame ground
2	Transmit Data (TD)	Data transmit line
3	Receive Data (RD)	Data receive line
4	Request To Send (RTS)	RTS control
5	Clear To Send (CTS)	CTS control
6	Data Set Ready (DSR)	Not used
7	Signal Ground (SG)	Signal ground
8–19	Not connected	—
20	Data Terminal Ready (DTR)	High when E7000 power is on
21–25	Not connected	—

B.3 Printer Connector

The layouts of the connector pin at the E7000 end of the provided printer cable is shown in figure B.3. Signal names and pin assignments are given in table B.3.

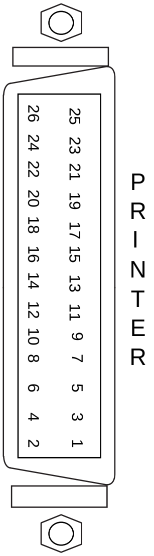


Figure B.3 Printer Cable Connector Pin Layouts

Table B.3 Printer Cable Connector Pin Assignment

Pin No.	Signal Name	Pin No.	Signal Name
1	$\overline{\text{FAULT}}$	14	GND
2	±0 V	15	$\overline{\text{ACKNLG}}$
3	$\overline{\text{DATA}}$ $\overline{\text{STROBE}}$	16	GND
4	GND	17	BUSY
5	DATA1	18	GND
6	DATA2	19	PE
7	DATA3	20	GND
8	DATA4	21	SELECT
9	GND	22	GND
10	DATA5	23	$\overline{\text{INPUT}}$ $\overline{\text{PRIME}}$
11	DATA6	24	GND
12	DATA7	25	NC
13	DATA8	26	NC

Signal Functions: Signal functions listed in table B.3 are described below.

DATA1 to DATA8: Data is output by these lines.

$\overline{\text{DATA}}$ $\overline{\text{STROBE}}$: Data is output from the data output lines when this signal goes low.

BUSY: The E7000 will not send the next data as long as this signal is high.

$\overline{\text{ACKNLG}}$: This signal goes low and the E7000 outputs the next data.

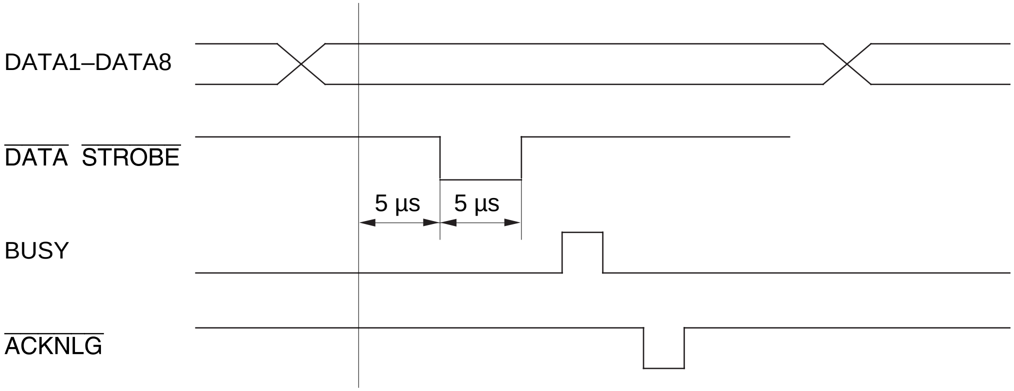
PE: If this signal goes high, the E7000 stops data outputs and displays the error message
***8: PAPER EMPTY.

SELECT: If this signal is low, the E7000 sends no data and outputs the error message
***7: PRINTER NOT READY.

$\overline{\text{FAULT}}$: If this signal is low, the E7000 sends no data and outputs an error message. If the PE signal is high, the E7000 outputs the error message
***8: PAPER EMPTY
If the PE signal is low, the E7000 outputs the error message
***7: PRINTER NOT READY

$\overline{\text{INPUT}}$ $\overline{\text{PRIME}}$: The E7000 forces this signal low when it starts up.

Data Output Timing: Data output timing of the E7000 is shown in figure B.4.


DATA1–DATA8
$\overline{\text{DATA}}$ $\overline{\text{STROBE}}$
5 µs
5 µs
BUSY
$\overline{\text{ACKNLG}}$

Note: Since the emulator checks the BUSY and $\overline{\text{ACKNLG}}$ signals, make sure that they are connected in the printer interface cable.

Figure B.4 Data Output Timing

B.4 LAN Connector

Figure B.5 shows pin locations in the LAN connector. Table B.4 lists pin numbers and signal names.

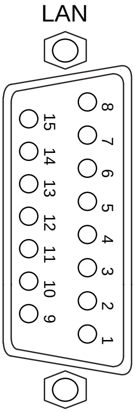


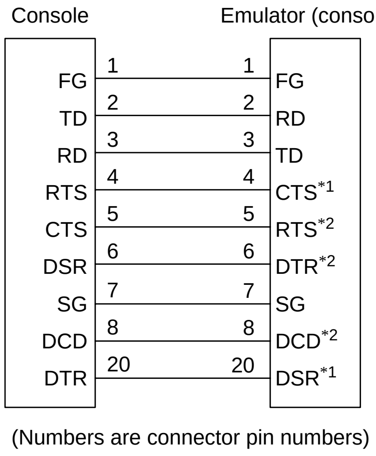
Figure B.5 LAN Connector Pin Locations

Table B.4 Pin Numbers and Signal Names in LAN Connector

Pin	Signal Name
1	NC
2	COL+
3	TX+
4	—
5	RX+
6	GND
7	—
8	—
9	COL–
10	TX–
11	—
12	RX–
13	+12 V
14	—
15	—

B.5 E7000 to Console Connection

Figure B.6 shows the wiring for the console connection. A console is connected to the console connector on the emulator station rear panel with the provided console interface cable.



Notes: 1. There is no need to connect these pins since they are not used by the emulator.

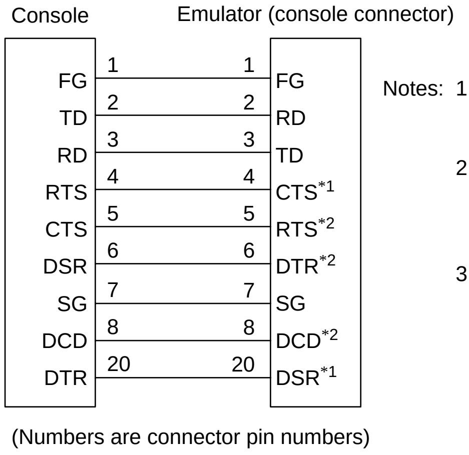
2. High when emulator’s power is on. There is no need to connect these pins if they are not used at the console end.

3. Pins 9–19, 21, and the following are not connected.

Figure B.6 Console to E7000 Wiring

B.6 E7000 to Host System Connection

Figure B.7 shows the wiring for the E7000 to host system connection when the optional host system interface cable is used.



Notes: 1. There is no need to connect these pins since they are not used by the emulator.

2. High when emulator’s power is on. There is no need to connect these pins if they are not used at the console end.

3. Pins 9–19, 21, and the following are not connected.

Figure B.7 Host System to E7000 Wiring

Note that provided host system interface cable may not be suitable for some host systems. In that case, use the wiring shown in figure B.8.

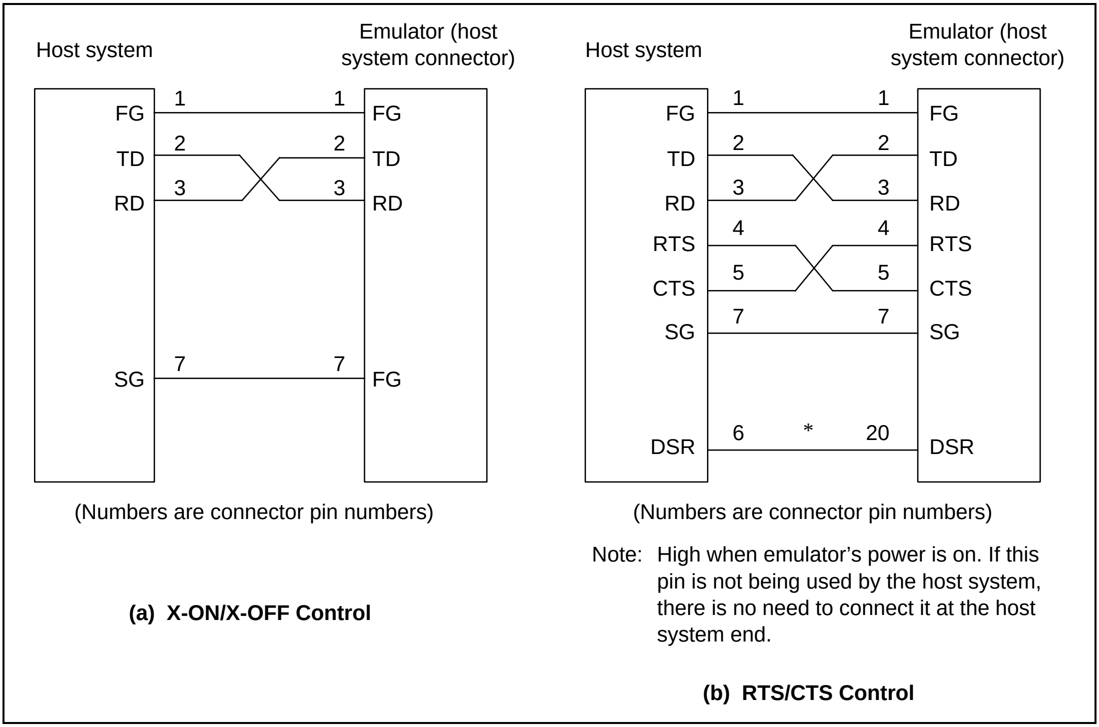

Host system
Emulator (host system connector)
1 FG — 1 FG
2 TD — 2 TD
3 RD — 3 RD
7 SG — 7 FG
(Numbers are connector pin numbers)
(a) X-ON/X-OFF Control
Host system
Emulator (host system connector)
1 FG — 1 FG
2 TD — 2 TD
3 RD — 3 RD
4 RTS — 4 RTS
5 CTS — 5 CTS
7 SG — 7 SG
6 DSR — * — 20 DSR
(Numbers are connector pin numbers)
Note: High when emulator’s power is on. If this pin is not being used by the host system, there is no need to connect it at the host system end.
(b) RTS/CTS Control

Figure B.8 Host System Wiring (Using Other Cable)

B.7 Printer Cable Connection

Table B.5 shows the signal names and their corresponding printer and E7000 pin numbers.

Table B.5 Pin Numbers and Signal Names in Printer and E7000

Pin No.	Signal Name	Printer	E7000	Remarks
1	$\overline{\text{DATA}}$ $\overline{\text{STROBE}}$ GND	1 19	3 4	
2	DATA1 GND	2 20	5 4	
3	DATA2 GND	3 21	6 16	
4	DATA3 GND	4 22	7 9	
5	DATA4 GND	5 23	8 9	
6	DATA5 GND	6 24	10 18	
7	DATA6 GND	7 25	11 20	
8	DATA7 GND	8 26	12 14	
9	DATA8 GND	9 27	13 14	
10	$\overline{\text{ACKNLG}}$ GND	10 28	15 16	
11	BUSY GND	11 29	17 18	
12	PE GND	12 29	19 20	
13	SELECT GND	13 20	21 22	
14	$\overline{\text{INPUT}}$ $\overline{\text{PRIME}}$ GND	31 30	23 24	
15	$\overline{\text{FAULT}}$ ±0 V	32 16	1 2	
16	FG FG			The cable shield and line pairs 16, 17, and 18 are all held to the frame ground. At the printer end, they are all connected to pin 17.
17	FG FG			
18	FG FG			

Part II　E7000PC Guide

Section 1 Overview

This system is an efficient software and hardware development support tool for application systems using the SH7604 (abbreviated to SH-2) microcomputer developed by Hitachi, Ltd. The SH-2 MCU contains the following components on a single chip:

- High-speed CPU
- Cache
- Timers
- Serial communication interface
- Refresh controller
- DMAC

When the E7000PC is connected to a user system, it operates in place of the SH-2 MCU and performs realtime emulation of the user system. Additionally, the E7000PC provides functions for efficient software and hardware debugging.

The E7000PC consists of an emulator station and emulator pod, as shown in figure 1.1. The emulator pod should be directly connected to the user system.

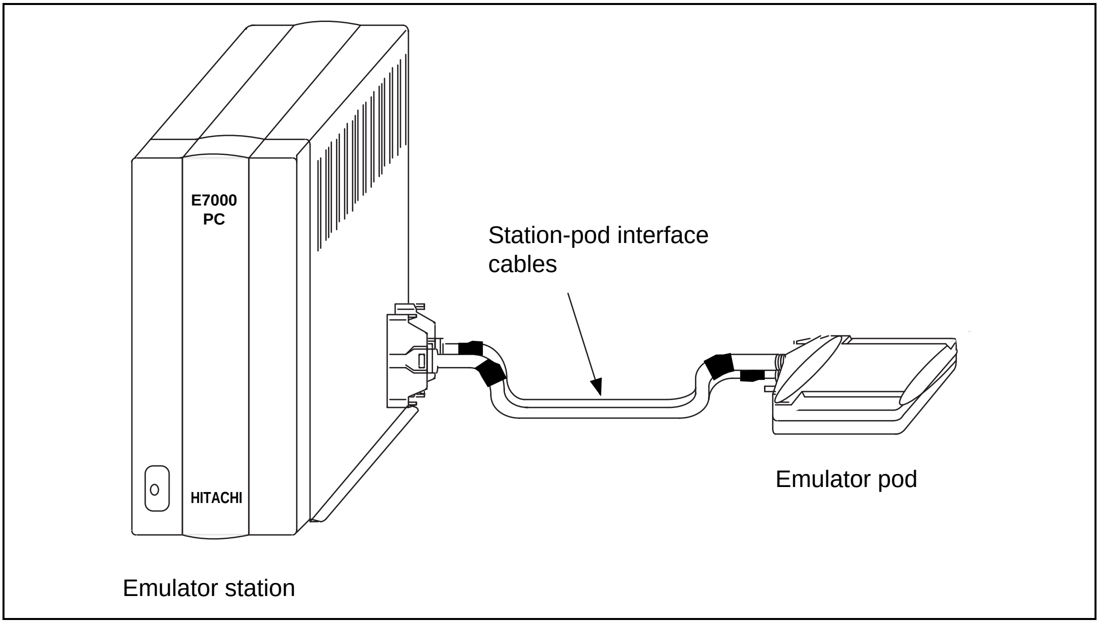


Figure 1.1 SH-2 E7000PC Emulator

The E7000PC provides the following features:

- Realtime emulation of SH-2
- A wide selection of emulation commands, promoting efficient system development
- Help functions to facilitate command usage without a manual
- Efficient debugging enabled by variable break functions and mass-storage trace memory (32 kcycles)
- Command execution during emulation, for example:
  - Trace data display
  - Emulation memory display and modification
- Measurement of subroutine execution time and frequency for evaluating the execution efficiency of user programs
- An IBM PC board for interfacing with an IBM PC, enabling high-speed downloading (1 Mbyte/min) of user programs
- E7000PC graphical user interface software (E7000PC GUI: option) can be loaded into the workstation to enable:
  - Graphic display operations in a multi-window environment
  - Source level debugging
  - Graphic display of trace information
- 512-kbyte standard emulation memory as substitute user system memory

1.1 Warnings

Before using the E7000PC, carefully read the following warnings. If the E7000PC is not used correctly, breakdowns may occur.

1. Check all components with the component list after unpacking the E7000PC.
2. Never place heavy objects on the casing.
3. Observe the following conditions in the area where the E7000PC is to be used:
    - Make sure that the internal cooling fans on the sides of the emulator station are at least 20 cm (8") away from walls or other equipment.
    - Keep out of direct sunlight or heat. Refer to section 1.2, Environmental Conditions.
    - Use in an environment with constant temperature and humidity.
    - Protect the E7000PC from dust.
    - Avoid subjecting the E7000PC to excessive vibration. Refer to section 1.2, Environmental Conditions.
4. Protect the E7000PC from excessive impacts and stresses.
5. Before using the E7000PC’s power supply, check its specifications such as power output, voltage, and frequency. For details on power supply, refer to section 1.2, Environmental Conditions.
6. When moving the E7000PC, take care not to vibrate or otherwise damage it. Pay special attention to exposed parts such as the power switch and I/O connectors.
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 E7000PC emulator and connected parts after connecting all cables. Cables should not be connected or removed when the power is on.
9. For details on differences between the SH-2 microcomputer and the E7000PC, refer to section 2, Differences between the SH-2 and the Emulator in Part III, Emulator Function Guide.
10. The optional 1-Mbyte or 4-Mbyte emulation memory board and the bus monitor board for the E7000 emulator cannot be used in this E7000PC emulator. Check that neither the emulation memory board nor the bus monitor board is installed in the E7000PC emulator station. If installed, remove the boards from the emulator station before connecting the emulator pod.

1.2 Environmental Conditions

Observe the conditions listed in table 1.1 when using the E7000PC emulator.

Table 1.1 Environmental Conditions

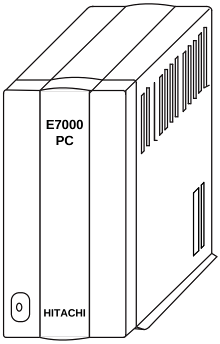
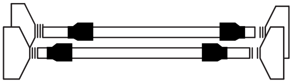
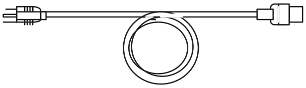
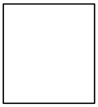
Item	Specifications
Temperature	Operating: +10 to +35°C Storage: –10 to +50°C
Humidity	Operating: 35 to 80% RH (no condensation) Storage: 35 to 80% RH (no condensation)
Vibration	Operating: 2.45 m/s^2 max. Storage: 4.9 m/s^2 max. Transportation: 14.7 m/s^2 max.
AC input power	Voltage: 100/200 VAC ±10% Frequency: 50/60 Hz Power consumption: 200 VA
Ambient gases	Must be no corrosive gases

1.3 Components

The E7000PC emulator consists of the emulator station and emulator pod. Check all the components after unpacking.

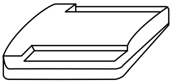
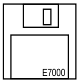
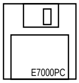
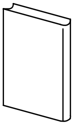
1.3.1 E7000PC Emulator Station

Table 1.2 E7000PC Emulator Station Components

Item	Configuration		Quantity	Remarks
Hardware	E7000PC emulator station		1	Power supply, control board, and trace board
	Station-pod interface cables		2	50 cm
	AC power cable		1	
	Fuse		1	Spare (3 A)
Documen-tation	HS7000ESTP1H Description Notes		1	HS7000ESTP1HE

1.3.2 E7000PC Emulator Pod

Table 1.3 E7000PC Emulator Pod Components

Item	Configuration		Quantity	Remarks
Hardware	Emulator pod		1	Fitted with two boards
Software	Floppy disks		1	E7000 system program (cannot be used with E7000PC emulator)
			1	E7000PC/IBM PC system program
Documen-tation	SH7604 E7000 Emulator User's Manual		1	HS7604EPD70HE

1.3.3 Option

In addition to the emulator station and pod components, the option listed in table 1.4 is also available. Refer to the option manual for details on the optional component.

Table 1.4 Optional Component Specifications

Item	Model Name	Specifications
IBM PC interface board	HS7000EII01H	• AT-bus specifications • Interface cable
Description Notes on Using the IBM-PC Interface Board (HS7000EII01H) used for the E7000PC Emulator or the Compact Evaluation Board	HS7000EII01HE	

Section 2 Components

2.1 E7000PC Hardware Components

As shown in figure 2.1, the E7000PC emulator consists of an emulator station (having a PC interface), emulator pod, and IBM PC interface board.

Note: Optional emulation memory boards, bus monitor boards, and LAN boards cannot be installed in the E7000PC.

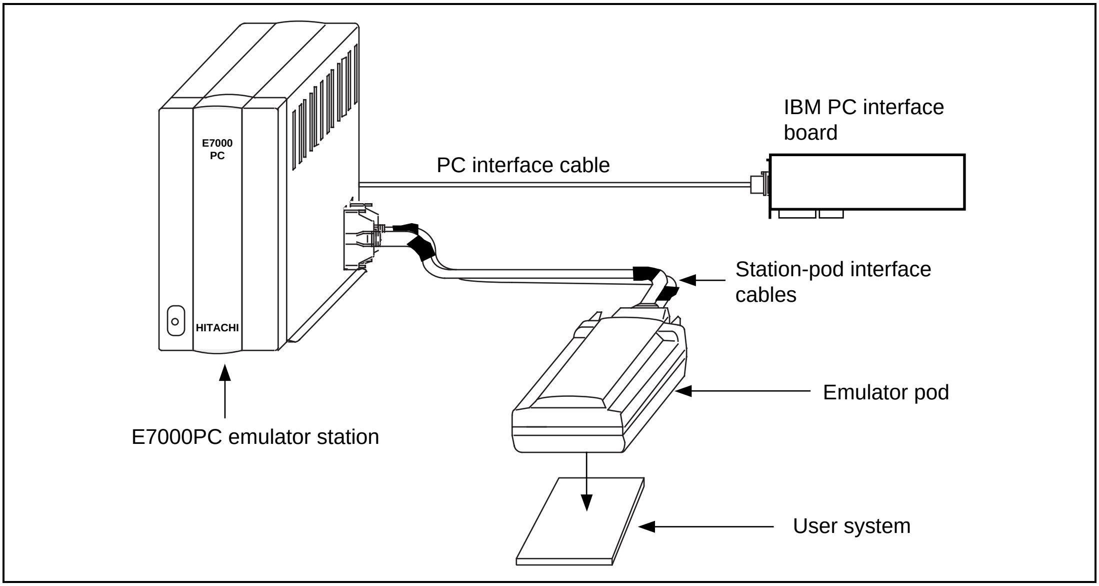


Figure 2.1 E7000PC Emulator Hardware Components

2.1.1 E7000PC Emulator Station Components

Front Panel:

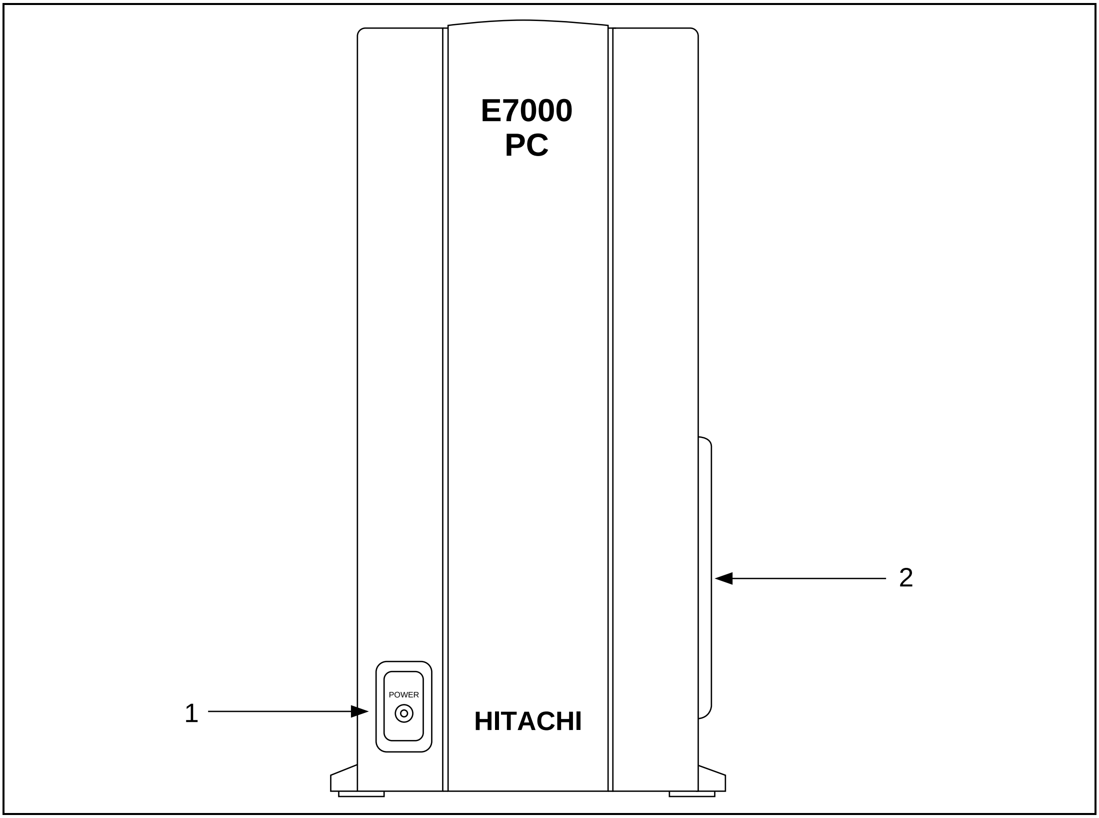


Figure 2.2 E7000PC Emulator Station Front Panel

1. Power-on lamp

   Lights when the E7000PC emulator power is on.

2. Station-pod interface cable connectors

   For connection to the E7000PC emulator station and pod.

Rear Panel:

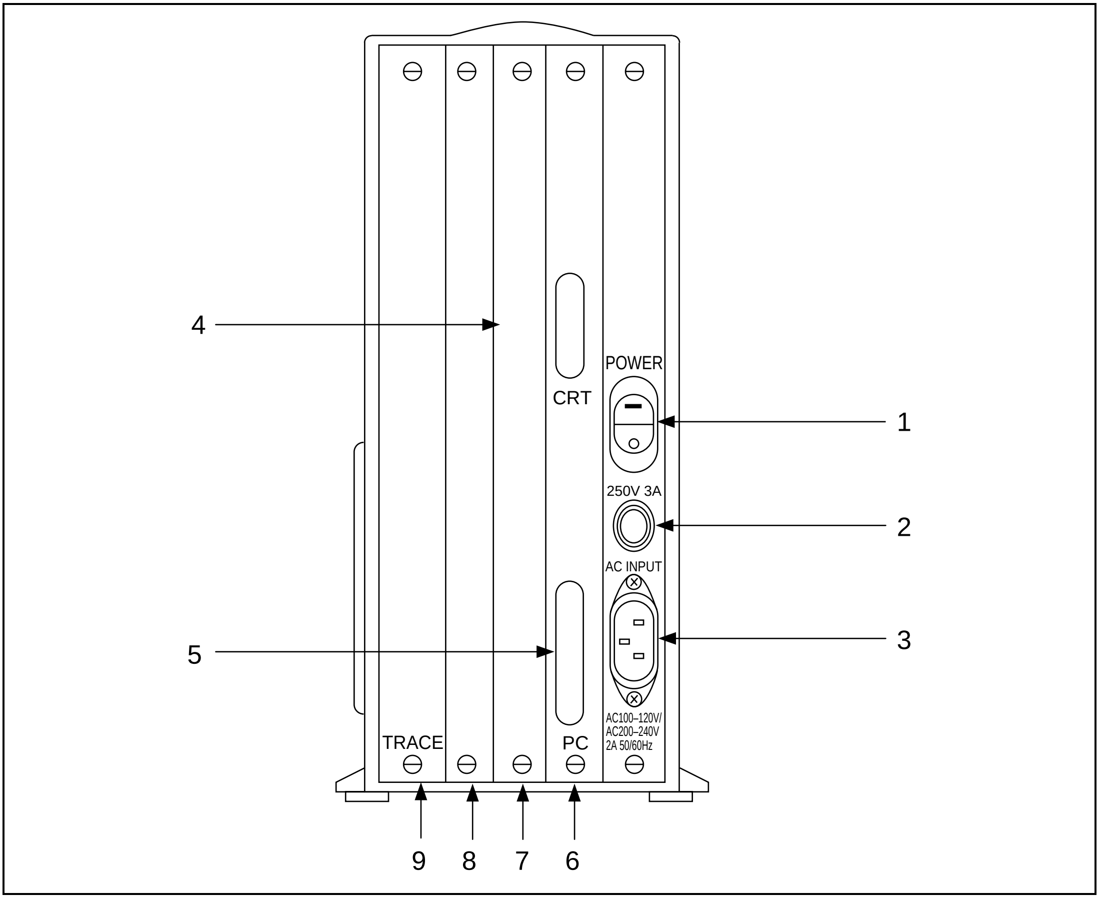


Figure 2.3 E7000PC Emulator Station Rear Panel

1. Power switch

   Turning this switch to I (input) supplies power to the E7000PC (emulator station and pod).

2. Fuse box

   Contains a 3-A 250 VAC fuse.

3. AC power connector

   For an 100/200 VAC power supply.

4. Console connector

   For future use. Marked CRT.

5. Personal computer connector

   For connection to the IBM PC console. Marked PC.

6. Control board slot

   For the control board.

7. Extension slot

   For system extension.

8. Extension slot

   For system extension.

9. Trace board slot

   For installing the trace board.

2.1.2 E7000PC Emulator Pod Components

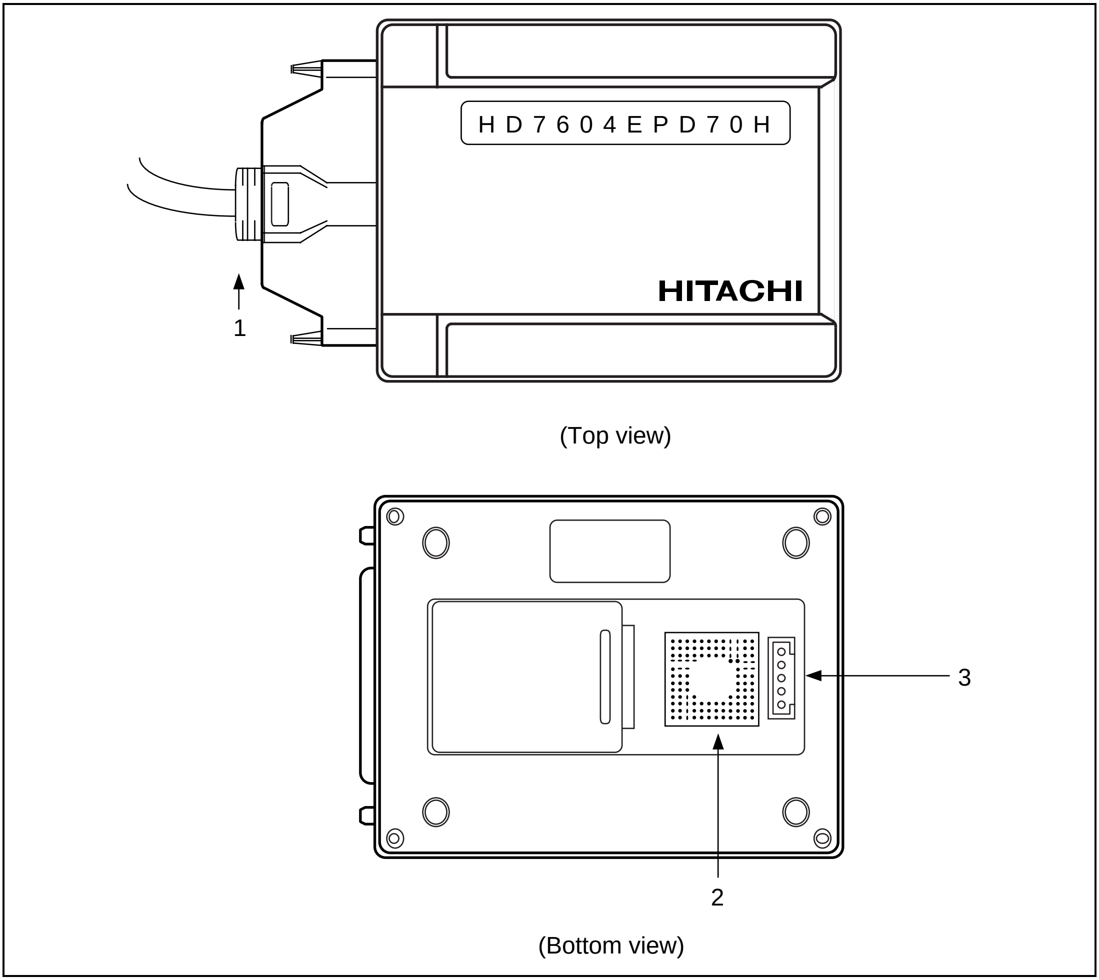


Figure 2.4 E7000PC Emulator Pod

1. Station-pod interface cables

   For connecting the E7000PC emulator station to the emulator pod.

2. User system connector

   For connection to the user system.

3. External probe connector

   For connection to the external probe.

2.2 E7000PC Software Components

The E7000PC emulator's software components are illustrated in figure 2.5. The emulator pod contains two 3.5-inch floppy disks; the E7000PC emulator system disk has "E7000PC" written under "HITACHI" on its label. The system disk files are described in table 2.1.

Table 2.1 Contents of E7000PC System Disk

File Name	Contents	Description
E7000.SYS	E7000PC system program	Controls the emulator pod and processes commands, such as emulation commands. Loaded into the emulator memory after the E7000PC system program is activated.
SHPOD764.SYS	SH-2 control program	Controls the MCU within the emulator pod. Loaded into the emulator memory after the E7000PC system program is activated.
SHCNF764.SYS	Configuration file	Contains MCU operating mode and MAP information. Loaded with the E7000PC system program.
LANCNF.SYS	LAN configuration file	Contains a host system name and IP address information when the E7000PC is connected to the workstation via the LAN interface. Cannot be used for the E7000PC.
IPI. EXE	Interface software	Executes on an IBM PC to interface with the E7000PC.
DIAG.TM	Diagnostic program	Loaded into the emulator station memory for testing and maintenance.

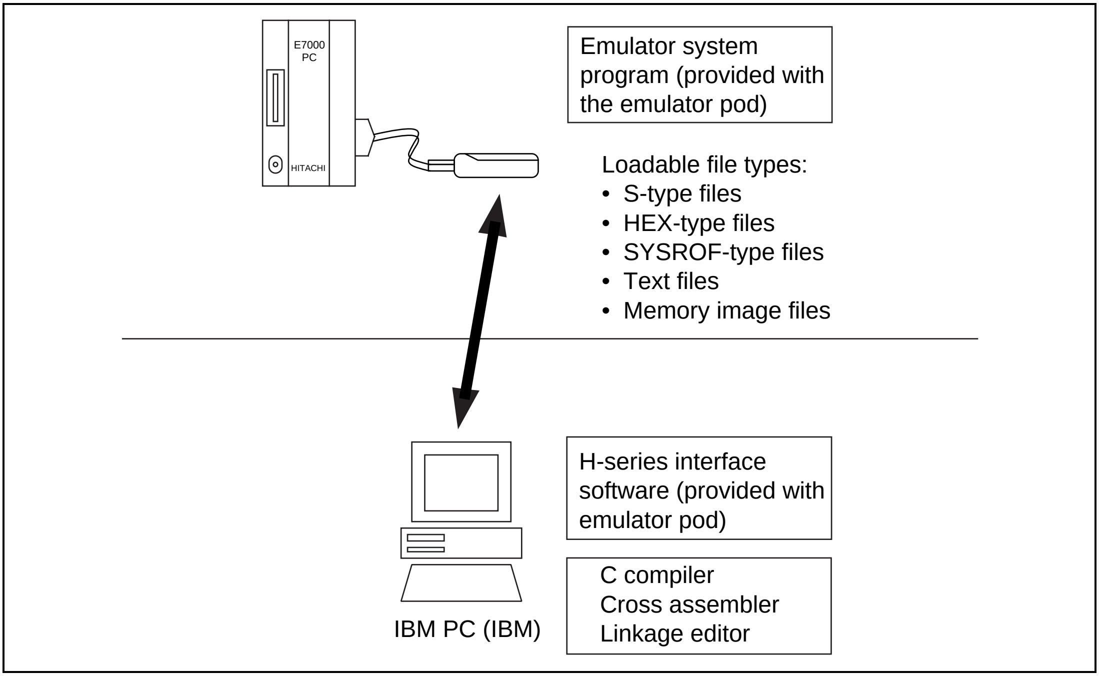


Figure 2.5 E7000PC Emulator Software Components

2.3 System Configuration

By installing an IBM PC interface board in the personal computer conforming to IBM PC AT-bus specifications, the E7000PC can be connected to the personal computer through the interface cable supplied with the IBM PC interface board. The system configuration is shown in figure 2.6.

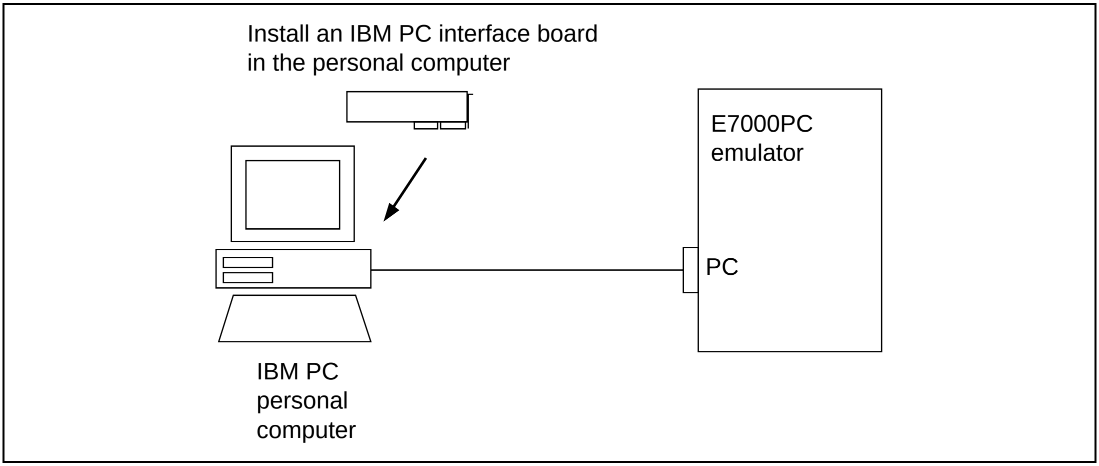
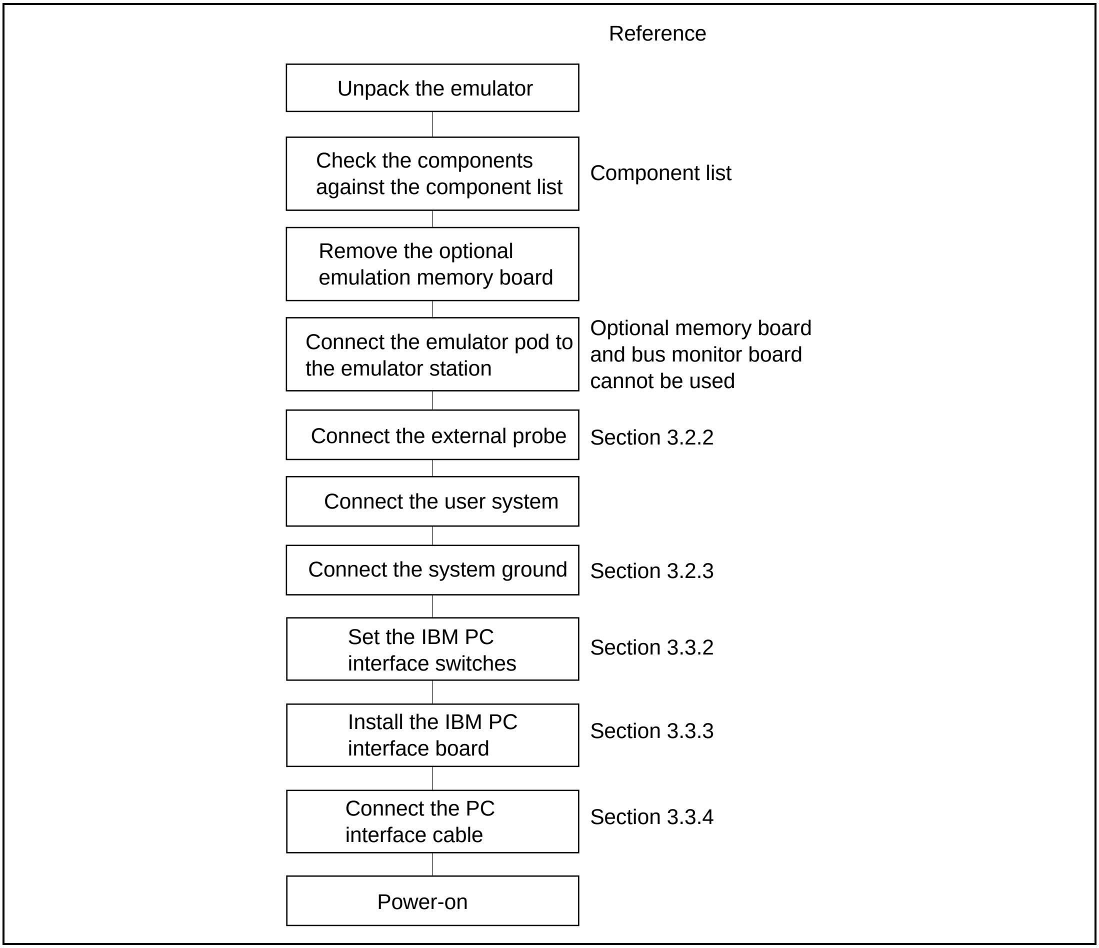


Figure 2.6 E7000PC Emulator System Configuration

Section 3 Preparation before Use

3.1 E7000PC Preparation

Unpack the E7000PC and prepare it for use as follows:



Reference
Unpack the emulator
Check the components against the component list
Component list
Remove the optional emulation memory board
Connect the emulator pod to the emulator station
Optional memory board and bus monitor board cannot be used
Connect the external probe
Section 3.2.2
Connect the user system
Connect the system ground
Section 3.2.3
Set the IBM PC interface switches
Section 3.3.2
Install the IBM PC interface board
Section 3.3.3
Connect the PC interface cable
Section 3.3.4
Power-on

Figure 3.1 E7000PC Preparation Flow Chart

3.2 E7000PC Connection

3.2.1 Connecting Emulator Pod

The emulator pod and the emulator station are packed separately. Use the following procedure to connect the emulator pod to the emulator station, or to disconnect it when moving the E7000PC:

(1) Check that the E7000PC power is off by ensuring that the power lamp on the left side of the emulator station front panel is extinguished.

(2) Remove the AC power cable for the emulator station from the outlet.

(3) Connect station-pod interface cables P1 and P2 to station-pod interface connectors J1 and J2 on the right side of the emulator station, respectively. Insert the longer screw of each cable to the connector screw hole without a spacer, and the shorter screw to the hole with a spacer. Tighten the longer screw first until the shorter screw reaches the spacer, then alternately tighten the longer and shorter screws. Figure 3.2 shows how to connect the station-pod interface cables to the emulator station.

Note: When connecting the cables, prevent the upper or lower side of the cables from lifting off the connector. Tighten the screws and push the cables gradually toward the connector.

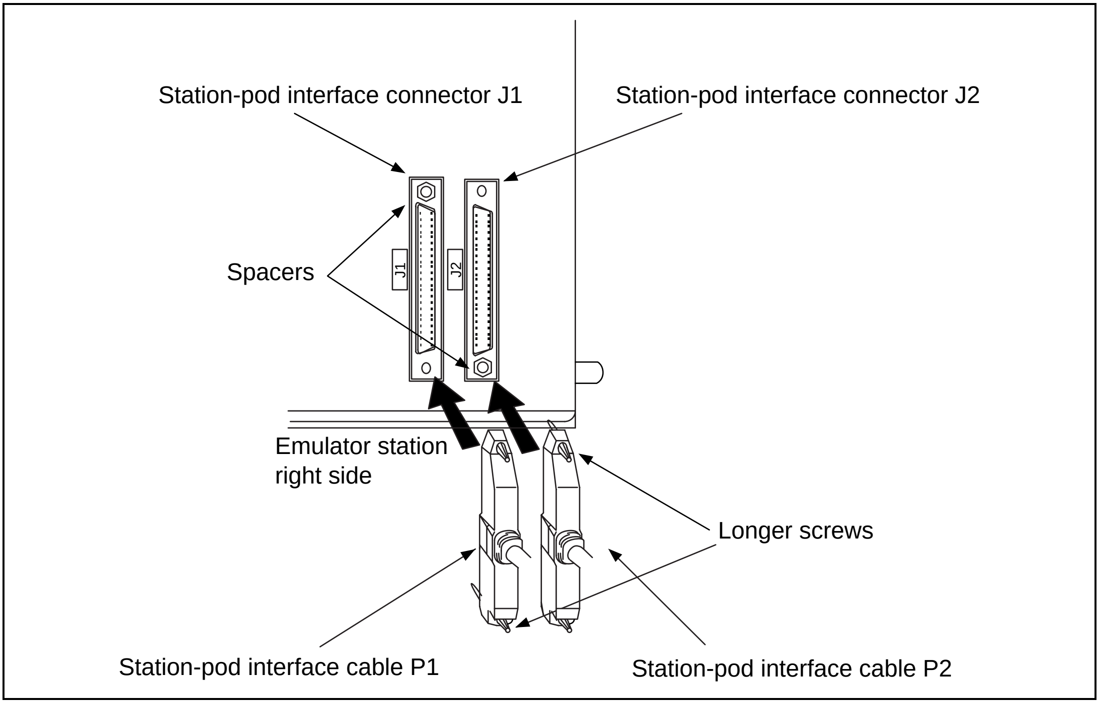

Station-pod interface connector J1
Station-pod interface connector J2
Spacers
J1
J2
Emulator station
right side
Longer screws
Station-pod interface cable P1
Station-pod interface cable P2

Figure 3.2 Connecting Station-Pod Interface Cables to Emulator Station

(4) Connect station-pod interface cables P1 and P2 to station-pod interface connectors J1 and J2, respectively, in the same way as connection to the emulator station. Tighten the screws in the same way as in step (3). See figure 3.3 for details.

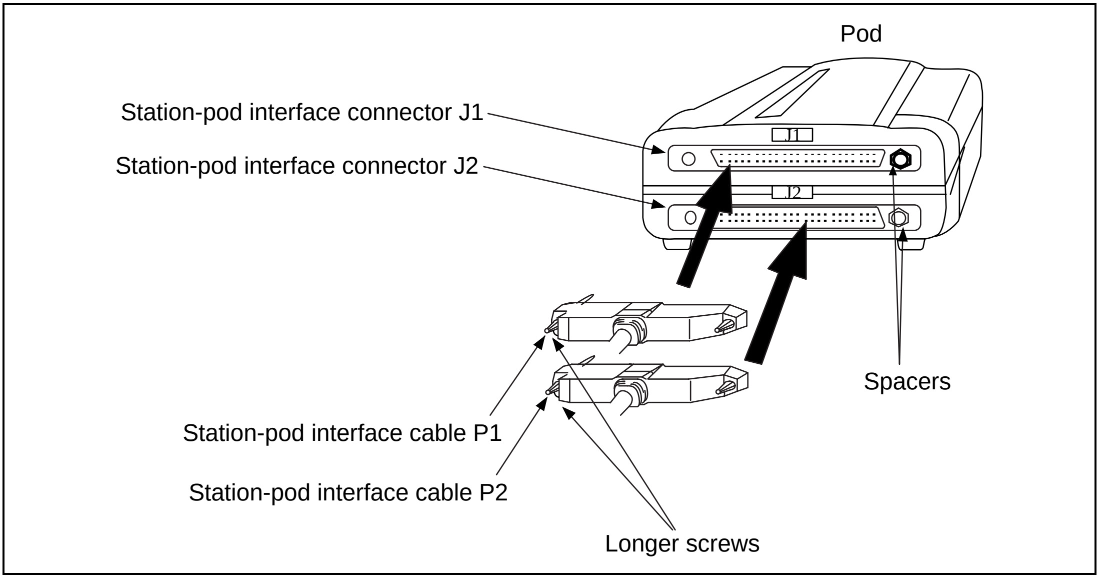
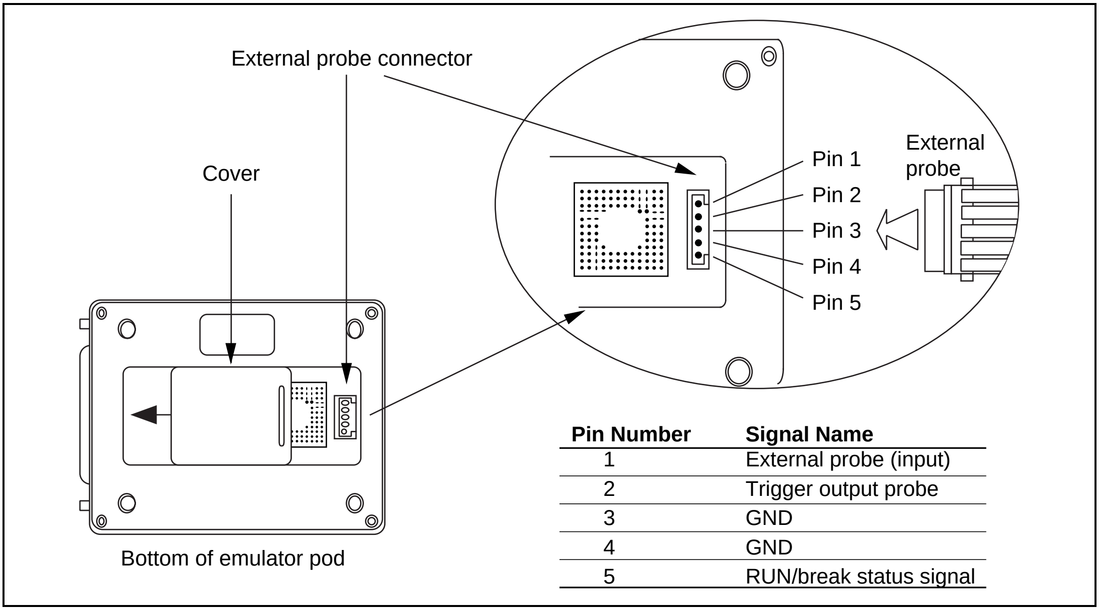

Pod
Station-pod interface connector J1
Station-pod interface connector J2
J1
J2
Spacers
Station-pod interface cable P1
Station-pod interface cable P2
Longer screws

Figure 3.3 Connecting Station-Pod Interface Cables to Emulator Pod

3.2.2 External Probe Connector

When an external probe is connected to the emulator pod, it enables external signal trace and multibreak detection. Figure 3.4 shows the external probe connector.



Pin Number	Signal Name
1	External probe (input)
2	Trigger output probe
3	GND
4	GND
5	RUN/break status signal

Figure 3.4 Connecting External Probe

3.2.3 Connecting System Ground

The E7000PC’s signal ground is connected to the user system’s signal ground via the emulator pod. In the emulator station, the signal ground and the frame ground are connected (figure 3.5). At the user system, connect the frame ground only; do not connect the signal ground to the frame ground. If it is difficult to separate the signal ground from the frame ground, ground the user system at the same outlet as the E7000PC’s power supply (figure 3.6).

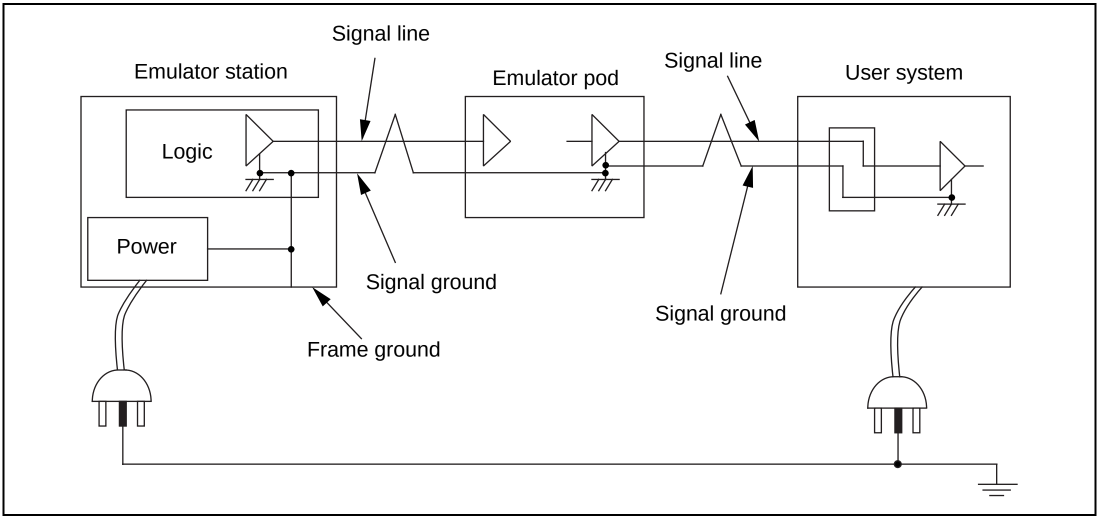


Figure 3.5 Connecting System Ground

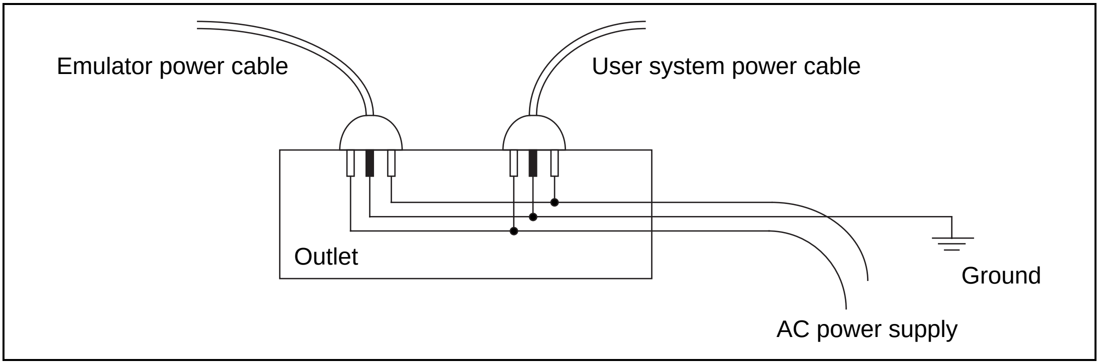


Figure 3.6 Connecting Frame Ground

The user system must be connected to an appropriate ground so as to minimize noise, ground loops, and other adverse effects. Confirm that the ground pins of the emulator pod are firmly connected to the user system’s ground.

3.3 System Connection

This section describes how to connect the E7000PC to an IBM PC via an IBM PC interface board.

3.3.1 IBM PC Interface Board Specifications

Table 3.1 IBM PC Interface Board Specifications

Item	Specification
Target personal computer	IBM PC*[1] conforming to an AT bus or compatible computer
System bus	AT bus
Memory requirement	16 kbytes
Memory allocation	By switches Memory for the interface board can be allocated within the address range from H'A0000 to H'FFFFF at any 16-kbyte boundary.
Interrupt	One interrupt must be selected from IRQ03, IRQ05, IRQ11, and IRQ12; unnecessary, however, if not used by application software*[2].
Interrupt selection	By switches
I/O area	No I/O area for this IBM PC interface board

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

2. In this manual, application software refers to software such as graphical user interface software and IBM PC interface software that uses the IBM PC interface board.

3.3.2 Setting the Switches on the IBM PC Interface Board

Allocating the Memory Area: The IBM PC interface board uses 16 kbytes of memory on the IBM PC. This memory must be allocated to a memory area on the IBM PC using switches on the IBM PC interface board. In particular, it can be allocated to any 16-kbyte block within the address range H'A0000 to H'FFFFF (figure 3.7). Note that the allocated memory area must not overlap memory already allocated to other boards. At shipment, the memory area of the IBM PC interface board is allocated to the address range from H'D0000 to H'D3FFFF.

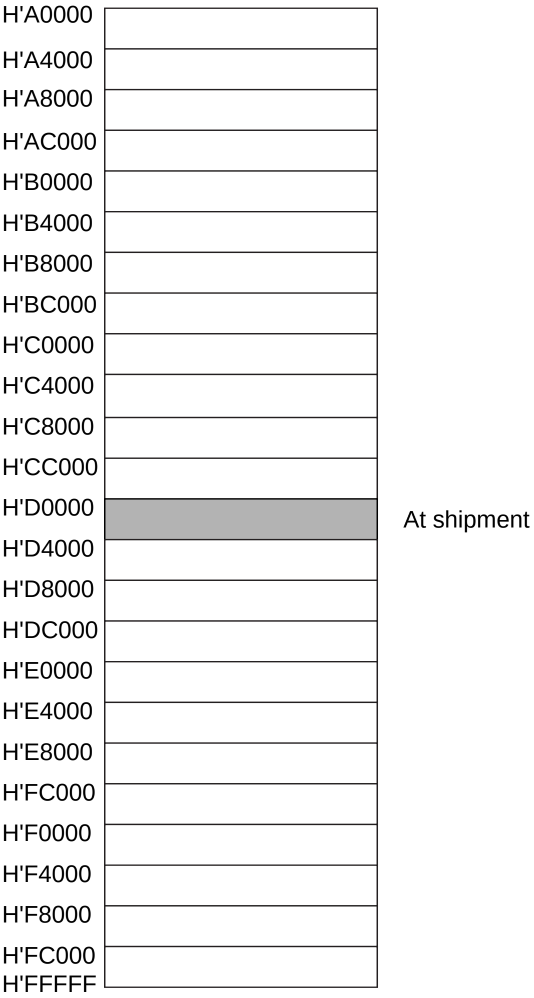


Figure 3.7 Memory Areas Allocatable for the IBM PC Interface Board

Selecting an Interrupt: One of four IBM PC interrupts can be selected by switches on the IBM PC interface board for application software that uses IBM PC interrupts. Make sure that application software for this board uses IBM PC interrupts before setting the switches. Available interrupts are listed in table 3.2. Select one interrupt in table 3.2 which is not used for other boards on the IBM PC. IRQ11 is set at shipment.

Table 3.2 Available Interrupts

Interrupt Level	Remarks
IRQ11	At shipment
IRQ12	
IRQ03	
IRQ05	

Setting the Switches: Eight switches are provided on the IBM PC interface board to allocate memory and select an interrupt. Figure 3.8 shows how to set these switches. The switch select conditions are listed in tables 3.3 and 3.4. Switch 8 is reserved for future use and must be closed.

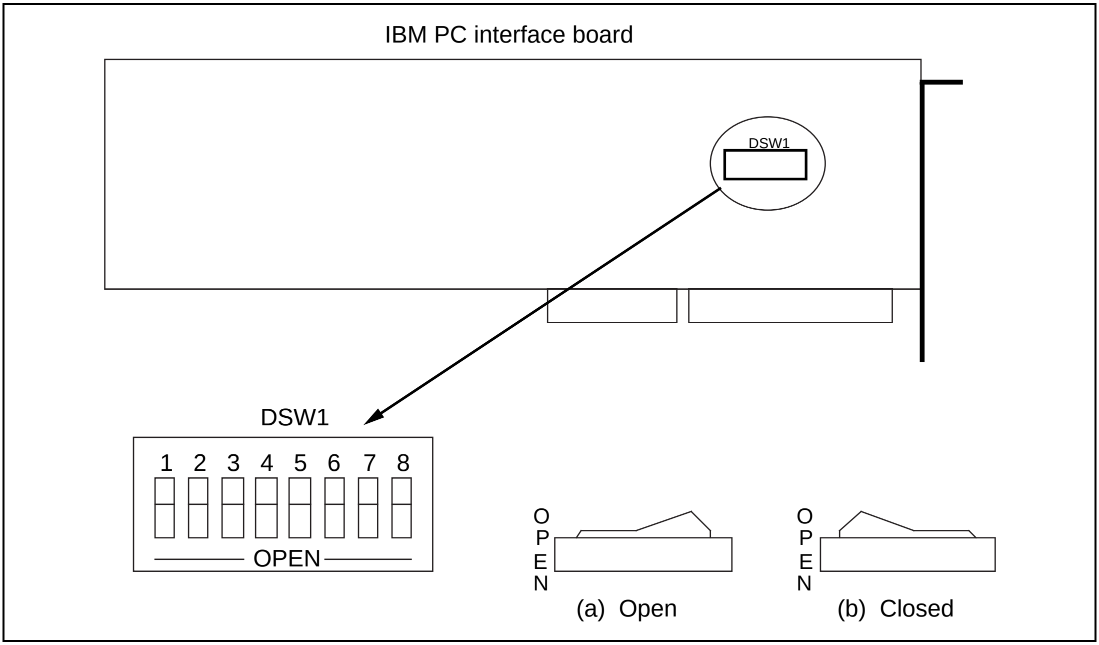


Figure 3.8 Switches on the IBM PC Interface Board

Table 3.3 Memory Allocation and Switch Settings

	Switch Settings					
IBM PC Address Range	**1**	**2**	**3**	**4**	**5**	**Remarks**
H'A0000 to H'A3FFF	Closed	Closed	Closed	Open	Closed	
H'A4000 to H'A7FFF	Open	Closed	Closed	Open	Closed	
H'A8000 to H'ABFFF	Closed	Open	Closed	Open	Closed	
H'AC000 to H'AFFFF	Open	Open	Closed	Open	Closed	
H'B0000 to H'B3FFF	Closed	Closed	Open	Open	Closed	
H'B4000 to H'B7FFF	Open	Closed	Open	Open	Closed	
H'B8000 to H'BBFFF	Closed	Open	Open	Open	Closed	
H'BC000 to H'BFFFF	Open	Open	Open	Open	Closed	
H'C0000 to H'C3FFF	Closed	Closed	Closed	Closed	Open	
H'C4000 to H'C7FFF	Open	Closed	Closed	Closed	Open	
H'C8000 to H'CBFFF	Closed	Open	Closed	Closed	Open	
H'CC000 to H'CFFFF	Open	Open	Closed	Closed	Open	
H'D0000 to H'D3FFF	Closed	Closed	Open	Closed	Open	At shipment
H'D4000 to H'D7FFF	Open	Closed	Open	Closed	Open	
H'D8000 to H'DBFFF	Closed	Open	Open	Closed	Open	
H'DC000 to H'DFFFF	Open	Open	Open	Closed	Open	
H'E0000 to H'E3FFF	Closed	Closed	Closed	Open	Open	
H'E4000 to H'E7FFF	Open	Closed	Closed	Open	Open	
H'E8000 to H'EBFFF	Closed	Open	Closed	Open	Open	
H'EC000 to H'EFFFF	Open	Open	Closed	Open	Open	
H'F0000 to H'F3FFF	Closed	Closed	Open	Open	Open	
H'F4000 to H'F7FFF	Open	Closed	Open	Open	Open	
H'F8000 to H'FBFFF	Closed	Open	Open	Open	Open	
H'FC000 to H'FFFFF	Open	Open	Open	Open	Open	

Table 3.4 Interrupts and Switch Settings

Interrupt	Switch 6	Switch 7	Remarks
IRQ11	Closed	Closed	At shipment
IRQ12	Closed	Open	
IRQ03	Open	Closed	
IRQ05	Open	Open	

Notes: 1. Set switches 6 and 7 closed (default at shipment) when interrupts are not used by application software that runs on the IBM PC interface board.

2. Switch 8 is reserved for future use and must be closed.

3.3.3 Installing the IBM PC Interface Board

Open the IBM PC cover and install the IBM PC interface board into an expansion slot conforming to AT bus specifications. Gently push the IBM PC interface board into the connector and fasten the board with the IBM PC screw.

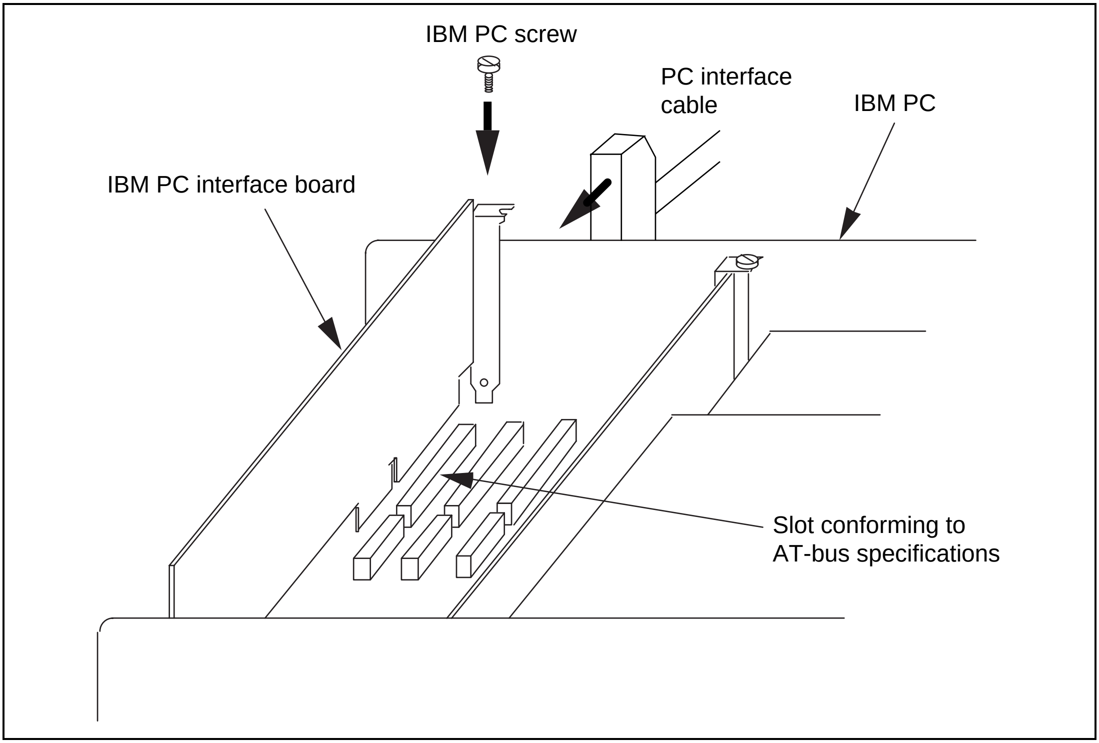


Figure 3.9 Installing IBM PC Interface Board

3.3.4 Connecting the IBM PC Interface Board to the E7000PC Emulator Station

To use the E7000PC emulator, connect the IBM PC interface board to the E7000PC emulator station via the supplied PC interface cable, as shown in figure 3.10.

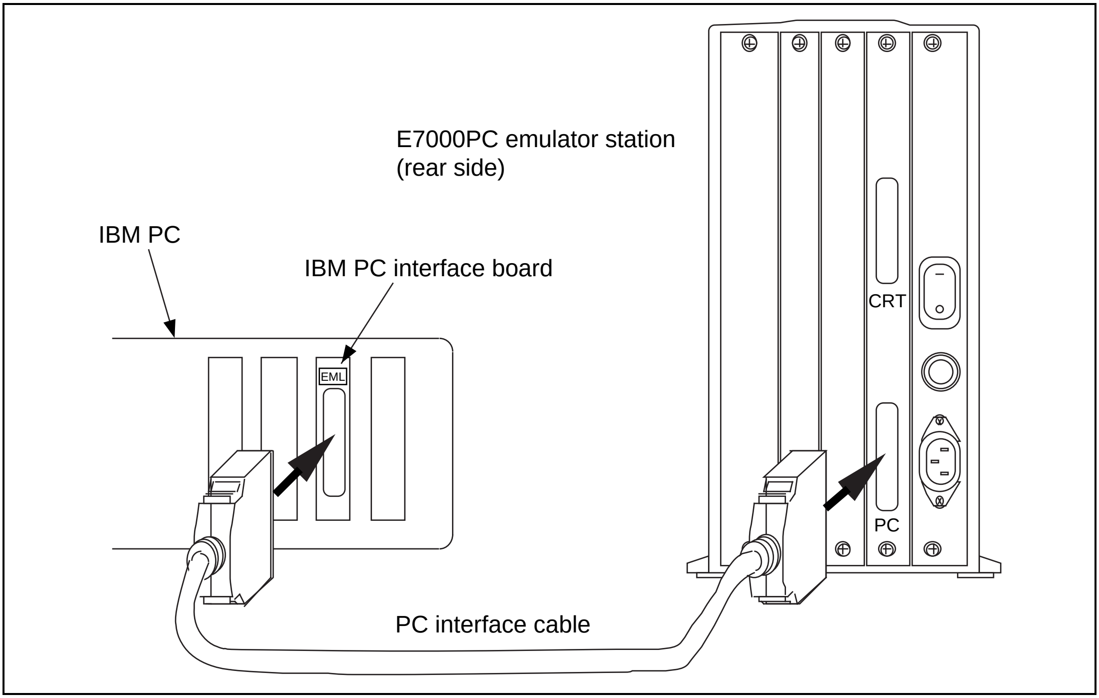


Figure 3.10 Connecting IBM PC Interface Board to E7000PC Emulator Station

3.4 System Software Installation

3.4.1 E7000PC System Software

The E7000PC system program must be installed through the IBM PC because the E7000PC does not have a floppy disk drive.

Two system disks are provided with the SH7604 emulator pod. Use the E7000PC system disk labelled as E7000PC under the HITACHI label (the one labelled as E7000 is for the E7000 and cannot be used for the E7000PC).

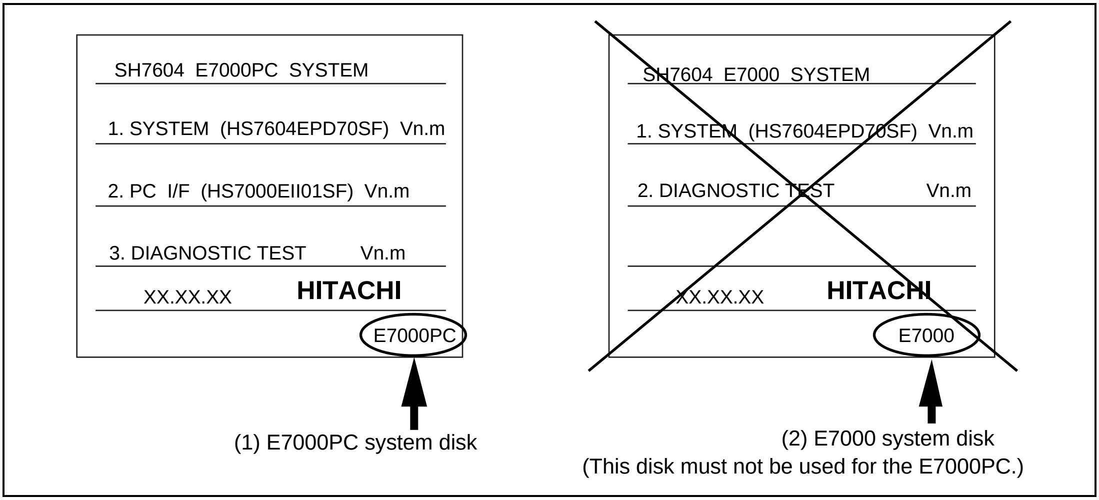


Figure 3.11 E7000PC System Disk

The E7000PC system disk is formatted as a 1.44-Mbyte disk on the IBM PC, and includes the following six files:

- E7000.SYS
- SHPOD764.SYS
- SHCNF764.SYS
- LANCNF.SYS
- IPI.EXE (IBM PC interface software)
- DIAG. TM

These files must be installed on the IBM PC.

To use the E7000PC, interface software must be running on the IBM PC. Two types of interface software are available: the above IBM PC interface software supplied with the SH7604 emulator pod, or the SH7604 E7000PC graphical user interface software (optional).

3.4.2 Installation

Copying E7000PC System Program: Copy all files in the E7000PC system disk to the IBM PC. Specify the copy destination directory in an environment variable when using the interface software*[1].

Setting Environment Variable: Before using the interface software, specify the interface software memory address in an environment variable. The most convenient way to do this is to add the following command to the file AUTOEXEC.BAT.

```
SET E7000SYS =[<directory name>][,[<termination code>] ,[<board address]]
         (a)                (b)                   (c)                   (d)
```

(a) The environment variable used by this interface software. The environment variable at default is E7000SYS*[2]. Input it using upper-case letters.

(b) To connect Hitachi’s E7000PC emulator using this interface software, input the name of the directory on which the E7000PC system program file exists. Input a back slash (\) to separate each directory name in the file path (do not input a back slash after the last directory name).

(c) Input a termination code for this interface software in hexadecimal. When omitted, the default value is H'1B (ESC).

(d) Input the eight highest-order bits (in hexadecimal) of the segment address of the memory where the interface board is installed. When the board address is omitted, the memory where the interface board is installed will be searched from the DIP switch settings shown in table 3.3.

Example: Input the following when the E7000PC system program file is in the directory C:\E7000\SH7030, the termination code is 04 (CTRL+D), and the interface board is installed at the address D8000:0000:

```
C>SET E7000SYS=C:\E7000\SH7030, 04, D8 (RET)
```

Copying Interface Software: Copy the interface software (IPI.EXE) and specify the path name for the directory.

Notes: 1. The E7000PC system disk contains this interface software (IPI.EXE). However, the interface software can be copied to another directory.

2. It is also possible to specify an environment variable with a name other than E7000SYS when starting up the IBM PC interface software (IPI).

Example:

```
C:\>SET E7000SYS=C:\E7000,1B,D0 ....Specify the default value
C:\>SET E7000=C:\E7000\SH,1B,D4 .....Specify E7000 as the environment variable
C:\>IPI E7000...........................................Initiate the IPI
                                    (with E7000 as the environment variable)
```

```
C:\>IPI ......................................................Initiate the IPI
                                                              (with the default environment variable)
```

Modifying CONFIG.SYS File: If the virtual EMS driver (EMM386.EXE) is used, the memory address range of the virtual EMS driver and that of the IBM PC interface board must not overlap. To prevent this, the CONFIG.SYS file contents must be modified as follows:

Specify the memory address range of the IBM PC interface board from D000:0000 to D3FF:0000 and the page frame start address of the virtual EMS driver E000:0000.

Example:

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

Notes: 1. Be sure to buck up the CONFIG.SYS file before modifying it.

2. How to specify addresses and address ranges varies depending on an IBM PC system software used. For more details, refer to the IBM PC manual.

Modifying SYSTEM.INI File: To initiate the interface software from Windows, first activate "MS-DOS Prompt" in the main group of the "Program Manager" window, then input the appropriate command.

The memory address range of the IBM PC interface board (D000:0000 to D3FF:0000) must be outside the Windows' memory control range. The SYSTEM.INI file contents, which is the initialization file for the Windows, must be modified as follows:

Specify EMMExclude in section [386Enh] of the SYSTEM.INI file by entering the underlined section shown below.

Example:

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

Notes: 1. Modifying the SYSTEM.INI file is not necessary if the Windows is not used.

2. How to specify addresses and address ranges varies depending on an IBM PC system software used. For more details, refer to the IBM PC manual.

3. Be sure to buck up the SYSTEM.INI file before modifying it.

4. The SYSTEM.INI file is located in the same directory where the Windows file exists.

3.5 Power-On Procedure for the E7000PC

Figure 3.12 shows the E7000PC power-on procedure when the RS-232C interface is used.

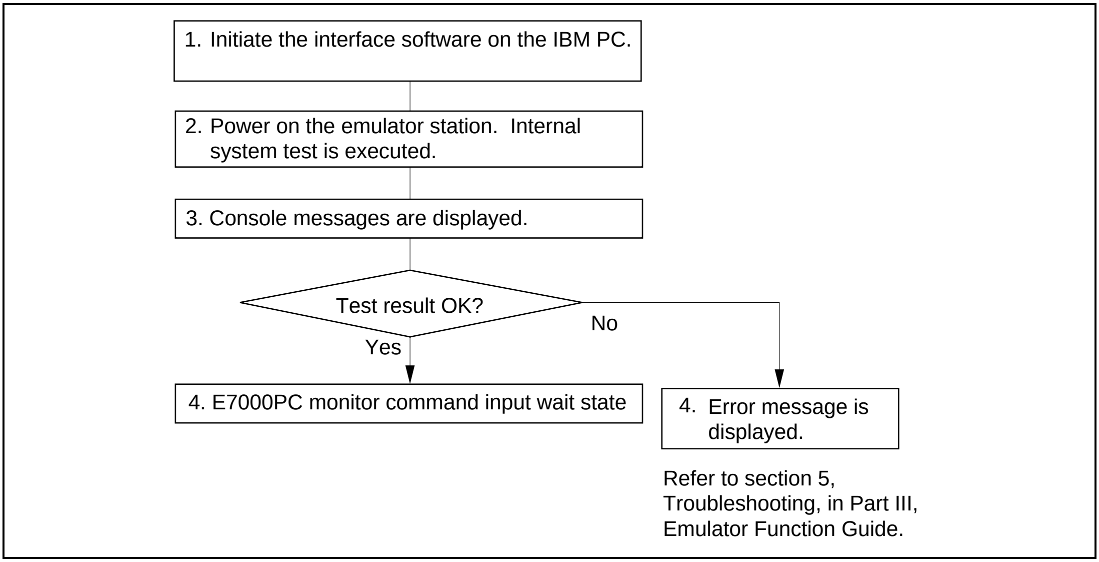

1. Initiate the interface software on the IBM PC.
2. Power on the emulator station. Internal system test is executed.
3. Console messages are displayed.
Test result OK?
Yes
No
4. E7000PC monitor command input wait state
4. Error message is displayed.
Refer to section 5, Troubleshooting, in Part III, Emulator Function Guide.

Figure 3.12 Power-On Procedure for the E7000PC

For initiating the interface software, refer to section 3.7.1, Initiating Interface Software.

For operations after power-on, refer to section 3.6.1, E7000PC Monitor Initiation and section 3.6.2, E7000PC System Program Initiation.

3.6 E7000PC Monitor Commands

3.6.1 E7000PC Monitor Initiation

When the E7000PC is turned on and the interface software is activated, the following messages are displayed.

Console Messages:

```
E7000 MONITOR Vn.m
Copyright (C) 1993 Hitachi, Ltd.                    (a)
Licensed Material of Hitachi, Ltd.

TESTING                                             (b)
RAM 0123

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

Descriptions:

(a) E7000PC monitor start message. Vn.m is the E7000PC monitor's version number. If this message is not displayed, determine what is wrong by reading section 5, Troubleshooting in Part III, Emulator Function Guide.

(b) The E7000PC internal system is being tested. A number from 0 to 3 is displayed when each of the four MCU internal RAM blocks has been tested. If an error occurs, the following messages are displayed:

```
*** RAM ERROR ADDR = xxxxxxxx  W-DATA = xxxxxxxx  R-DATA = xxxxxxxx
*** xxxxx REGISTER ERROR  W-DATA = xxxx  R-DATA = xxxx
```

If these messages are displayed, refer to section 5, Troubleshooting in Part III, Emulator Function Guide.

(c) List of E7000PC monitor commands. Enter the required command at the cursor position. These commands are described in table 3.5. When an L or T command is specified, the E7000PC will prompt for another command after execution is completed. After the system program has been loaded by an S or R command, the QUIT command ends the system program execution and returns the E7000PC monitor to command input wait state.

Table 3.5 E7000PC Monitor Commands

Command	Function	Reference Section
S	E7000PC system program initiation Initiates the system program. When the system program has not been loaded, this command loads it from the floppy disk and then initiates the system.	Section 3.6.2, E7000PC System Program Initiation
R	E7000PC system program reload Loads and initiates a different system program from the loaded system program.	Section 3.6.2, E7000PC System Program Initiation
L	IP address display Displays the E7000PC IP address.	This command is reserved for future use.
T	Diagnostic program initiation Loads and initiates the diagnostic program in the E7000PC system floppy disk. If a problem occurs, use this command to initiate the diagnostic program.	Attached diagnostic program manual

3.6.2 E7000PC System Program Initiation

The E7000PC system program must be loaded and initiated before initiating the E7000PC.

If S or R is entered, followed by (RET), when the E7000PC is in monitor command input wait state, the E7000PC system program is loaded from the IBM PC and initiated.

Table 3.6 E7000PC System Program Initiation Commands

Command	Description
S	Loads and initiates the system program from the IBM PC. If the E7000PC system program is already loaded, the system program is initiated immediately.*
R	Reloads and initiates the E7000PC system program.

Note: This situation occurs when the system program is initiated and then terminated with the QUIT command. However, if the E7000PC monitor T (diagnostic test) command has been executed, or when the system program has been forcibly terminated by a clock error, the system program is reloaded.

Display at E7000PC System Program Initiation

```
START E7000                                              ┐
  S: START E7000                                         │
  R: RELOAD & START E7000                                │
  L: DISPLAY LAN PARAMETER                               │ (a)
  T: START DIAGNOSTIC TEST                               │
     (S/R/L/T) ?  { S } (RET)                            │
                  { R }                                  ┘

**  E7000 SYSTEM LOADING **                              ┘ (b)

SH7604 E7000 (HSxxxx EPDxxSF) Vn.m                       ┐
Copyright (C) Hitachi, Ltd. 1993                         │ (c)
Licensed Material of Hitachi, Ltd.                       ┘

CONFIGURATION FILE LOADING                               ┘ (d)
LAN IP ADDRESS FILE LOADING                              ┘ (e)
MODE CHECK                                               ┘ (f)
HARD WARE REGISTER READ/WRITE CHECK                      ┘ (g)
POD SYSTEM LOADING                                       ┘ (h)
EMULATOR POD TEST                                        ┘ (i)
**  RESET IN BY E7000  !                                 ┘ (j)
CLOCK = xxxx                                             ┘ (k)
MASTER  MODE = xx (MD 5-0 = xx)                          ┐ (l)
MODE  SET = x                                            ┘
REMAINS EMULATION MEMORY  S = D'xxxxxkB                  ┘ (m)

WARM OR COLD START                                       ┐
  file name          : WARM START                        │
  return             : COLD START                        │ (n)
  (file name/return) ? { <file name> (RET) }             │
                       { (RET)             }             ┘

:                                                        ┘ (o)
```

Description

(a) E7000PC command input request message. Enter S. Enter R if loading another E7000PC system program.

(b) The E7000PC system program is being loaded from the IBM PC.

(c) Start message of the E7000PC system program. Vn.m is the version number.

(d) Configuration file is being loaded from the IBM PC.

(e) IP address file for the LAN is being loaded from the IBM PC. (N.B.: The LAN cannot be used by the E7000PC.)

(f) The specified E7000PC mode is compared with the mode selection pin status on the user system. If the E7000PC is in master mode and the mode selection pin status is in slave mode, the SH-2 operating mode cannot be set to the mode specified in the configuration file. In this case, initiation will be aborted and the following message will be displayed:

```
USER MODE = SLAVE
```

Either change the E7000PC mode to slave mode with the MODE command or change the mode selection pin status to master mode.

(g) Emulator station hardware test start message. If there is an error in the emulator station, an error message is displayed.

(h) The program to be executed in the emulator pod is being loaded from the IBM PC.

(i) Emulator pod test start message. If there is an error in the emulator pod, an error message is displayed.

(j) A RES signal has been input to the SH-2.

(k) Specified clock. If the user system is ready, the user system’s clock is used. If the user system’s clock is not ready but the 6.144-MHz E7000PC internal clock is ready, the E7000PC internal clock is used.

(l) SH-2 clock operating mode and user system mode selection pin state. The operating mode, the CS0 area bus width, and master/slave are those previously set with the MODE command (saved in configuration file). For details, refer to section 7.2.25, MODE in Part III, Emulator Function Guide.

(m) Remaining emulation memory size

(n) Specify either WARM START[*1] or COLD START[*2] as follows:

WARM START: Specify the file name containing recovery information.
COLD START: Press the (RET) key.

(o) E7000PC system program prompt. An E7000PC system program command can now be entered.

Notes: 1. WARM START recovers the information saved in a file when the E7000PC system program was terminated by the QUIT command. (For details, refer to section 7.2.30 QUIT in Part III, Emulator Function Guide.) The recovery information is listed below.

- Software breakpoints
- Hardware break conditions, trace stop conditions, and trace acquisition conditions
- Memory map information
- Configuration information
- Symbol information

2. COLD START initializes the above emulation information.

3.7 Interface Software Operations

3.7.1 Initiating Interface Software

The IBM PC interface software is initiated by inputting the IPI command. An example is shown below.

```
C>IPI<CR>                                                      (a)

H-SERIES PC INTERFACE (HS7000EII01SF) Ver n.m
Copyright(c) Hitachi, Ltd. 1993                                (b)
Licensed Material of Hitachi, Ltd.

INTERFACE BOARD ADDRESS=yyyy:zzzz,TERMINATE CODE=tt            (c)

(E7000PC commands can be entered here)                         (d)

<Termination key>                                              (e)

C>
```

Description:

(a) Initiate the interface software.

(b) The interface software start-up message is displayed.
Ver n.m represents the interface software version number.

(c) yyyy indicates the segment, zzzz indicates the offset, and tt indicates the interface software terminate code.

(d) The E7000PC operations can start here. When the E7000PC power is turned on, the E7000PC will enter the state described in section 3.6, E7000PC Monitor Commands.

(e) Enter the termination key to terminate the interface software and return the E7000PC to the IBM PC command input wait state. For more details on the termination key, refer to 3.4.2, Installation.

3.7.2 Emulation Support Function

This interface software provides two emulation support functions: automatic command input from IBM PC files, and logging acquisition of the console output to IBM PC files or a printer. These functions can only be invoked when the E7000PC is in the command input wait state, not in the data input wait state. Examples of the command input wait state and the data input wait state are shown below.

- Command Input Wait State

```
:
#
```

The E7000PC is in the command input wait state when it provides either of these prompts.

- Data Input Wait State (Memory Command)

```
:MEMORY 100(RET)
00000100 00     ? 11(RET)  ---Input data in the data input wait state
00000101 00     ?     --------Data input wait state
```

Automatic Command Input:

(1) To specify a command file (input file), input "<" and a filename without any space between the two as shown below.

Example:

```
: <FILENAME(RET)
```

(2) Commands from the specified command file will be automatically input and sent to the E7000PC. When the command file is specified as shown in the example below, the MAP, MEMORY, and CLOCK commands from the specified command file will be automatically executed. When additional data is required within a command as in the MEMORY command, this data will also be automatically input.

Example:

File contents:

```
MAP 0 FFFF;U
MEMORY 100
30
.
CLOCK
```

Execution results:

```
:<FILENAME(RET)
:MAP 0 FFFF;U
:MEMORY 100
 00000100 00  ? 30
 00000101 00  ? .
:CLOCK
 CLOCK=USER
: (Next command input wait state)
```

(3) Commands will be automatically input from the command file until the end of the file is reached or <CTRL + C> keys are input. Pressing <CTRL + C> keys forcibly terminates command execution and displays the following message, asking whether or not the automatic command input should be continued.

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

(a): Input Y to terminate or N to continue automatic command input.

(4) Logging acquisition can be specified from the command file. For details, refer to the description on logging in this section.

(5) If the interface software displays the following confirmation message during command execution, it will only accept a reply.

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

(a): Input Y to overwrite the existing file or N to terminate the transfer.

(6) Command files cannot be nested.

Logging:

(1) To specify logging, the E7000PC must be in the command input wait state. Input “>” followed by a filename without any space between the two. Examples of starting and quitting logging are shown below.

Examples:

```
: >FILENAME(RET)       To overwrite a file
: >>FILENAME(RET)      To add to a file
: >-(RET)              To quit logging
```

(2) If logging acquisition is specified with this command, the subsequent command input, execution results, and error messages will be displayed on the console and output to the file with the specified filename. If “PRN” is specified for the filename, the output will be sent to the printer.

(3) The following message will be displayed if you attempt to overwrite an existing file.

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

(a): Input Y to overwrite the file or N to quit.

(4) The following messages will not be logged.

— The program counter during GO command execution
— Addresses during loading, saving, and verifying

3.7.3 Notes

(1) The MS-DOS may display an error message during file transfer. When the error message is displayed before file transfer starts, such as when no floppy has been inserted to the specified drive, when an access to an unformatted floppy disk is specified, or when writing on a write-protected floppy disk is specified, remove the cause of the error and then enter “R” to recover. When the error message is displayed during file transfer, file transfer may not be completed successfully even after recovery from the error (R) is specified.

Entering “A” to terminate the error processing terminates the interface software.

(2) Pressing the <BREAK>, <STOP>, or <CTRL+C> keys during file transfer forcibly terminates the file transfer. Pressing the termination key during file transfer has no meaning.

(3) If the save operation results in an error, the interface software displays the received data and waits for the E7000PC command input.

(4) When a file is written to the IBM PC disk, it is first created as a temporary file with filetype “$$$”, and will only be renamed with the specified filetype when it is closed normally. At this stage, the existing file, if any, will be erased. If a forced termination occurs while writing the file, the temporary file will be erased, and the existing file will be left behind.

(5) When the E7000PC is initiated before the interface software, the message issued by the E7000PC before the interface software initiation cannot be displayed.

Section 4 Operating Examples

Section 4.1, Basic Examples, and section 4.2, Application Examples, include explanations based on the following user program.

```
ADDR        CODE     LABEL     MNEMONIC     OPERAND
01001000    E00A               MOV          #0A,R0
01001002    E101               MOV          #01,R1
01001004    E201               MOV          #01,R2
01001006    D405               MOV.L        0100101C,R4
01001008    6323     !LOOP     MOV          R2,R3
0100100A    321C               ADD          R1,R2
0100100C    2426               MOV.L        R2,@-R4
0100100E    6133               MOV          R3,R1
01001010    70FF               ADD          #FF,R0
01001012    8800               CMP/EQ       #00,R0
01001014    8BF8               BF           !LOOP
01001016    0009               NOP
01001018    AFFE     !LABEL    BRA          !LABEL
0100101A    0009               NOP
0100101C    0F10               .DATA.W      0F10
0100101E    0000               .DATA.W      0000
```

These examples assume that the emulator station is connected to the host system (IBM PC) via the PC interface, and that the user program is downloaded from the host system to the E7000PC. Therefore, store the program in the host system before initiating the E7000PC.

Initiate the E7000PC by the following procedure:

Operations	**Console Message**
1. Input the IPI command to the IBM PC.	C> IPI (RET)

2. Turn on the power on the E7000PC emulator station, that the E7000PC system disk is installed. The console displays the message shown on the right when the PC interface program operates.

```
H-SERIES PC INTERFACE (HS7000EII01SF) Ver n. m
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.

INTERFACE BOARD ADDRESS=yyyy:zzzz,  TERMINATE CODE=tt

E7000 MONITOR Vn. m
Copyright (C) 1993 Hitachi, Ltd.
Licensed Material of Hitachi, Ltd.

TESTING
RAM 0123

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

3. Enter S and (RET) to start up the E7000PC system.

4.1 Basic Examples

4.1.1 Preparing for Connection of IBM PC

Before connecting the host system, specify the host system name and the IP address by the following procedure:

Operations

1. To start up the PC interface software, enter the IPI command at the IBM PC.
   The console displays the message shown on the right and the E7000PC enters the monitor command input wait state.

Console Message

```
C>IPI (RET)

H-SERIES PC INTERFACE (HS7000EII01SF) Ver n. m
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.

INTERFACE BOARD ADDRESS=yyyy:zzzz,  TERMINATE CODE=tt

E7000 MONITOR Vn. m
Copyright (C) 1993 Hitachi, Ltd.
Licensed Material of Hitachi, Ltd.

TESTING
RAM 0123

START E7000
  S : START E7000
  R : RELOAD & START E7000
  L : DISPLAY LAN PARAMETER
  T : START DIAGNOSTIC TEST
```

2. Enter S (RET) to re-initiate the system.

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

** E7000 SYSTEM LOADING **

SH7604 E7000 (HSxxxxEPDxxSF) Vn. m
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
LAN IP ADDRESS FILE LOADING
MODE CHECK
HARDWARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TEST
** RESET IN BY E7000 **

CLOCK = xxxx
MASTER MODE = xx (MD x-x = xx)
MODE SET = x
REMAINS EMULATION MEMORY    S=D'xxxxxkB

WARM OR COLD START
file name  : WARM START
return     : COLD START
```

3. Enter (RET).

```
(file name / return) ? (RET)
:_
```

4.1.2 Specifying the SH-2 Operating Mode

Specify the E7000PC operating mode and the SH-2 operating mode by the following procedure:

Operations / **Console Message**

1. Enter MODE;C (RET) to specify the E7000PC operating mode.

```
:MODE;C (RET)
```

2. The console displays the message shown on the right. To select mode 8 of the SH-2 and use the configuration file data to set the SH-2 operating mode, for example, enter 8 (RET) and C (RET).

```
E7000 MODE  (MD5-0) = xx ? _
MODE SET (C:CONFIGURATION/U:USER/
                    M:MASTER–SLAVE)=M ? _

E7000 MODE  (MD5-0) = xx ? 8 (RET)
MODE SET (C:CONFIGURATION/U:USER/
                    M:MASTER–SLAVE)=M ? C (RET)
```

3. After the above specification has been completed, the console asks if the mode settings should be stored in the configuration file. To store the mode settings, enter Y (RET). After that, the E7000PC operates in the mode specified above whenever initiated with this system disk.
   To correct a mis-typed mode number, return to step 3 above before entering Y (RET) and repeat the procedure.
   Remove the write protect from the system floppy disk before storing the mode settings in the configuration file.

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

CONFIGURATION WRITE OK ? (Y/N) ? Y (RET)
```

4. After the mode settings have been stored in the configuration file, the E7000PC system program automatically terminates.

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

5. Enter S (RET) to re-initiate the system program.

```
        (S/R/L/T) ? S (RET)
WARM OR COLD START
file name  : WARM START
return     : COLD START
(file name / return) ? _
```

6. Enter (RET).

```
(file name  / return) ? (RET)
```

4.1.3 Allocating Standard Emulation Memory and Specifying Attributes

In order to load the user program to memory, allocate the standard emulation memory in the pods by the following procedure:

Operations **Console Message**

Enter MAP 1000000 101FFFF;S (RET) to allocate the standard emulation memory to addresses H'1000000 to H'101FFFF. The console displays the message shown on the right, which indicates that the memory allocation has been completed.

```
:MAP 1000000 101FFFF;S (RET)

 REMAINS EMULATION MEMORY     S=D’ 0384kB
```

Enter MAP (RET) and the console displays the attributes of all the memory areas.

```
:MAP (RET)
 01000000 - 0101FFFF;S   21000000 - 2101FFFF;S
 INTERNAL I/O    =       E0000000 - FFFFFFFF
 REMAINS EMULATION MEMORY     S=D’ 0384kB
```

4.1.4 Executing Program

Execute the loaded program by the following procedure:

Operations **Console Message**

1. Enter .SP (RET) then FFFFFFC (RET) as the SP value to set the stack pointer (SP register) to H'FFFFFFC.
   The console then asks for the program counter value. Enter 1001000 (RET) as the program counter value. The console then asks for the status register value. In this example, other registers need not be set or changed, therefore, enter . (RET) to exit this interactive mode.

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

 R15(SP) = xxxxxxxx ? FFFFFFC (RET)
 PC      = xxxxxxxx ? _

 PC      = xxxxxxxx ? 1001000 (RET)
 SR     = xxxxxxxx:- - IIII - - - - ? _

 SR     = xxxxxxxx:- - IIII - - - - ? . (RET)
:_
```

Note: In interactive mode, entering only (RET) makes no change to the currently displayed item, and the next item is displayed. In the above example, entering only (RET) can complete the register modification procedure. The register value can also be directly input without using the interactive mode. For example, to set the stack pointer value directly, enter .SP FFFFFFC (RET).

2. Enter GO (RET) to execute the program from the address pointed by the PC. While the program is executed, the console displays the current program counter value (shown as xxxxxxxx on the right).

```
:GO (RET)
 ** PC = xxxxxxxx
```

3. Enter (BREAK) key or (CTRL) + C keys to terminate program execution. The console displays the contents of the program counter, the status register, the control register, and the general registers R0 to R15 at termination. RUN - TIME shows the duration of program execution from the GO command execution to (BREAK) or (CTRL) + C key input. BREAK KEY shows that the execution has been terminated because (BREAK) or (CTRL) + C was entered.

```
:(BREAK)
 PC = 01001018  SR = 000000F1:- - IIII - - - T
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000000  00000059  00000090  00000059
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’0000H:00M:01S:012345US
 +++:BREAK KEY
:_
```

4.1.5 Software Break

Program execution can be stopped at a particular address by setting a breakpoint as follows:

Operations **Console Message**

1. Enter BREAK 1001010 (RET) to terminate program execution immediately before the instruction at address H'1001010 in the program is executed.

```
:BREAK 1001010 (RET)
```

2. Restart program execution from address H'1001000. This can be done in two ways: one is to enter the start address directly, and the other is to first set the program counter to H'1001000, then enter GO, as described in section 4.1.4, Executing Program.

```
:GO 1001000 (RET)
 ** PC = xxxxxxxx
```

3. The program execution terminates immediately before the instruction at address H'1001010 is executed. The console displays the data shown on the right. The BREAK POINT shows that the program execution terminated because of a software breakpoint.

```
 PC = 00001010  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000009  00000001  00000002  00000001
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’0000H:00M:00S:000006US
 +++:BREAK POINT
:_
```

4.1.6 Single-Step Execution

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

Operations | **Console Message**

1. The program counter points to the next address to be executed when the program execution terminates in the example of section 4.1.5, Software Break. Here, entering STEP (RET) executes only one instruction, and the console displays the information as shown on the right. 01001012 CMP/EQ #00,R0 shows the executed address and mnemonic code, and STEP NORMAL END shows that the single-step execution has terminated.

```
:STEP (RET)




 PC = 01001014  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000009  00000001  00000002  00000001
 R8 - 15 00000000  00000000  00000000  00000000
 01001012                      CMP/EQ     #00, R0
 +++:STEP NORMAL END
:_
```

2. To repeat single-step execution, enter only (RET). This can be repeated until another command is executed.

```
:(RET)
 PC = 01001008  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000000  00000001  00000002  00000001
 R8 - 15 00000000  00000000  00000000  00000000
 01001014                       BF           !LOOP
 +++:STEP NORMAL END
:_
```

4.1.7 Setting Hardware Break Conditions

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

Operations — **Console Message**

1. Enter BREAK - (RET) to cancel the breakpoint set in the example in section 4.1.5, Software Break.

```
: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'F0FFFF8, enter BREAK_CONDITION1 A = F0FFFF8 W (RET).

```
:BREAK_CONDITION1 A = F0FFFF8 W (RET)
```

4. Enter GO 1001000 (RET) to start executing the program from address H'1001000. When the break condition is satisfied, the console displays the information shown on the right. BREAK CONDITION1 shows that the program execution has terminated because the break condition was satisfied.

```
:GO 1001000 (RET)
 PC = 01001012  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000008  00000002  00000003  00000002
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’0000H:00M:00S:000010US
 +++:BREAK CONDITION1
:_
```

4.1.8 Displaying Trace Information

Trace information acquired during program execution can be displayed by the following procedure:

Operations **Console Message**

1. Enter TRACE (RET) to see the trace information. The console will display the instruction mnemonic information.

```
:TRACE (RET)

     IP          ADDR        LABEL     MNEMONIC     OPERAND
 *-D'00076    01001000                 MOV          #0A,R0
 *-D'00075    01001002                 MOV          #01,R1
 *-D'00074    01001004                 MOV          #01,R2
 *-D'00073    01001006                 MOV.L        0100101C,R4
 *-D'00072    01001008     !LOOP       MOV          R2,R3
      :           :                       :              :
```

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

```
:TRACE;B (RET)

     BP          AB          DB       MA   R/W  ST    IRL   NMI  RES  BRQ  PRB  VCC  CLK
 *           01001000                             MOV        #0A,  R0
   –D'00208  01001000  ******E0  EXT   R   PRG   1111   1    1    1    1    1    04
   –D'00207  01001001  ******0A  EXT   R   PRG   1111   1    1    1    1    1    02
 *           01001002                             MOV             #01,  R1
   –D'00206  01001002  ******E1  EXT   R   PRG   1111   1    1    1    1    1    02
   –D'00205  01001003  ******01  EXT   R   PRG   1111   1    1    1    1    1    02
 *           01001004                             MOV        #01,  R2
   –D'00204  01001004  ******E2  EXT   R   PRG   1111   1    1    1    1    1    02
   –D'00203  01001005  ******01  EXT   R   PRG   1111   1    1    1    1    1    02
                   :                              :                     :
```

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

```
(CTRL)+S
(CTRL)+Q
```

4.2 Application Examples

4.2.1 Break with Pass Count Condition

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

Operations **Console Message**

1. Enter BREAK 1001012 5 (RET) to terminate program execution when address H'1001012 is passed five times.

```
:BREAK 1001012 5 (RET)
```

2. To start execution from address H'1001000, enter GO 1001000 (RET).

```
:GO 1001000 (RET)
```

3. When execution terminates after address H'1001012 is passed five times, the console displays the data shown on the right.

```
 PC = 01001014  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D'0000H:00M:00S:000027US
 +++:BREAK POINT
:_
```

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

```
:BREAK (RET)
 ADDRESS      CNT        PASS       SYMBOL
 01001012     0005       0005
    (a)         (b)        (c)
```

4.2.2 Conditional Trace

The following procedure can be used to limit the acquisition of trace information during program execution.

Operations **Console Message**

1. To cancel the breakpoint set in the example of section 4.2.1, Break with Pass Count Condition, enter BREAK - (RET).

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

2. Enter TRACE_CONDITION A =1001000:1001014;R (RET) to get trace information only while the program counter is between addresses H'1001000 and H'1001014.

```
:TRACE_CONDITION A= 1001000:1001014;R (RET)
```

3. Enter GO 1001000 (RET) to start executing the program, then (BREAK) key or (CTRL) + C keys to terminate the execution.

```
:GO 1001000 (RET)
 ** PC = xxxxxxxx
 PC = 01001018  SR = xxxxxxxx:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D'0000H:00M:01S:000027US
 +++:BREAK KEY
:_
```

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

```
:TRACE (RET)

     IP        ADDR       LABEL      MNEMONIC     OPERAND
*-D'*****    01001000                MOV          #0A,R0
*-D'*****    01001002                MOV          #01,R1
*-D'*****    01001004                MOV          #01,R2
```

4.2.3 Parallel Mode

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

Operations **Console Message**

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

```
:GO 1001000 (RET)
 ** PC = xxxxxxxx (RET)
#_
```

2. Enter DUMP 1002000 100200F (RET) to display the memory contents from H'1002000 to H'100200F.

```
#DUMP 1002000 100200F (RET)

<ADDR>              <D    A    T    A>                               <ASCII CODE>
01002000    14 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     “. . . . . . . .”
```

3. Enter MEMORY 1001018 FC (RET) to modify the contents of memory address H'1001018 into FC.

```
#MEMORY 1001018 FC (RET)
```

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

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

5. To terminate program execution, enter (BREAK) key or (CTRL) + C keys.

```
(BREAK)
 PC = 01001018  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’ 0000H:01M:01S:000027US
 +++:BREAK KEY
:_
```

6. Enter DISASSEMBLE 1001000 100101F (RET) to confirm that the program has been changed by memory modification in parallel mode.

```
:DISASSEMBLE 1001000 100101F (RET)
```

```
ADDR        CODE      LABEL      MNEMONIC    OPERAND
01001000    E00A                 MOV         #0A,R0
01001002    E101                 MOV         #01,R1
01001004    E201                 MOV         #01,R2
01001006    D405                 MOV.L       0100101C,R4
01001008    6323      !LOOP      MOV         R2,R3
0100100A    321C                 ADD         R1,R2
0100100C    2426                 MOV.L       R2,@-R4
0100100E    6133                 MOV         R3,R1
01001010    70FF                 ADD         #FF,R0
01001012    8800                 CMP/EQ      #00,R0
01001014    8BFB                 BF          !LOOP
01001016    0009                 NOP
01001018    AFFC      !LABEL     BRA         1001016
0100101A    0009                 NOP
0100101C    0F10                 .DATA.W     0F10
0100101E    0000                 .DATA.W     0000
```

4.2.4 Searching Trace Information

The TRACE_SEARCH command can be used to search for a particular part of the acquired trace information.

Operations | **Console Message**

Enter TRACE_SEARCH A=1001018 (RET), and the console will only display those parts of the trace information in which the address bus value is H'1001018.

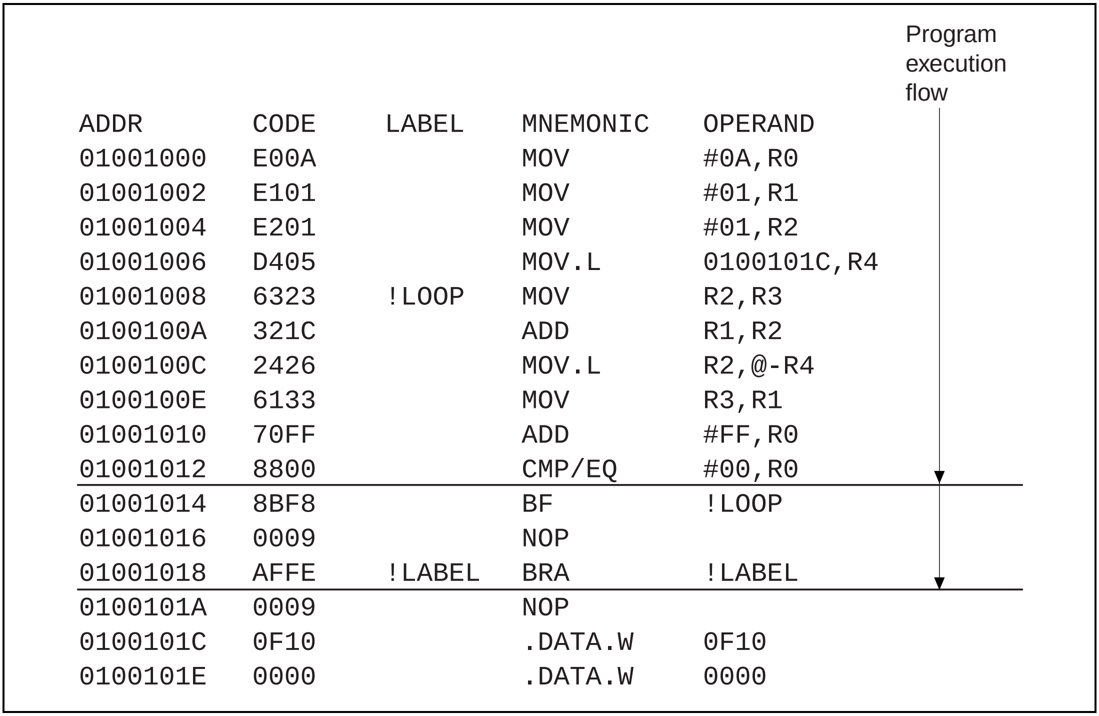
```
:TRACE_SEARCH A=1001018 (RET)
```

```
     BP          AB         DB        MA   R/W   ST    IRL   NMI   RES   BRQ   PRB  VCC   CLK
  -D'03392   01001018   00000000   EXT    R    PRG   1111   1     1     1     1     1    02
  -D'03879   01001018   00000000   EXT    R    PRG   1111   1     1     1     1     1    02
  -D'03766   01001018   00000000   EXT    R    PRG   1111   1     1     1     1     1    02
                  :                              :                            :
```

4.2.5 Sequential Software Break

A break can be generated when specified addresses are passed in a specified order, using the BREAK_SEQUENCE command as follows:

Operations	**Console Message**
1. Enter BREAK_SEQUENCE 1001012 1001018 (RET), which terminates program execution when the instructions at addresses H'1001012 and H'1001018 are executed consecutively in that order, as shown in figure 4.1.	:BREAK_SEQUENCE 1001012 1001018 (RET)



```
ADDR        CODE    LABEL    MNEMONIC    OPERAND
01001000    E00A             MOV         #0A,R0
01001002    E101             MOV         #01,R1
01001004    E201             MOV         #01,R2
01001006    D405             MOV.L       0100101C,R4
01001008    6323    !LOOP    MOV         R2,R3
0100100A    321C             ADD         R1,R2
0100100C    2426             MOV.L       R2,@-R4
0100100E    6133             MOV         R3,R1
01001010    70FF             ADD         #FF,R0
01001012    8800             CMP/EQ      #00,R0
01001014    8BF8             BF          !LOOP
01001016    0009             NOP
01001018    AFFE    !LABEL   BRA         !LABEL
0100101A    0009             NOP
0100101C    0F10             .DATA.W     0F10
0100101E    0000             .DATA.W     0000
```

Figure 4.1 Program Execution Flow

2. Enter GO 1001000 (RET) to execute the program. When the specified condition is satisfied, execution terminates, and the console displays the data shown on the right. The BREAK SEQUENCE shows that execution has terminated because the condition specified in the BREAK_SEQUENCE command has been satisfied.

```
:GO 1001000 (RET)
 PC = 01001018  SR = 000000F0:- - IIII - - - -
 PR = 00000000  GBR = 00000000  VBR = 00000000
 MACH = 00000000  MACL = 00000000
 R0 - 7   00000005  00000008  0000000D  00000008
 R8 - 15 00000000  00000000  00000000  00000000
 RUN - TIME = D’ 0000H:00M:00S:000027US
 +++:BREAK SEQUENCE
:_
```

3. Enter TRACE (RET) to confirm the executed instructions.

```
:TRACE (RET)
```

```
    IP          ADDR       LABEL     MNEMONIC     OPERAND
*-D'00005     0100100E               MOV          R3,R1
*-D'00004     01001010               ADD          #FF,R0
*-D'00003     01001012               CMP/EQ       #00,R0
*-D'00002     01001014               BF           !LOOP
*-D'00001     01001016               NOP
* D'00000     01001018   !LABEL      * BREAK *
:
```

Part III Emulator Function Guide

Section 1 Emulator Functions

1.1 Overview

This system is a hardware and software support tool for the development of systems incorporating SH7604 microcomputer (abbreviated to SH-2). In addition to a high-speed CPU, the SH-2 contains cache/internal RAM, timers, a serial communication interface, an interrupt controller, and a DMAC on the same chip. Table 1.1 shows the functions of the SH-2.

Table 1.1 SH-2 Functions

Function	Specifications	
Supported device	SH7604	
Maximum address	1 Gbyte	
Internal ROM	None	
Cache/internal RAM	4 kbytes	
Timer	16-bit free running timer: Watchdog timer:	1 channel 1 channel
Serial communication interface	Start-stop or clock synchronization:	1 channel
DMAC	4 channels	
Interrupt controller	External interrupts (NMI, IRL0 to IRL3): Internal interrupt sources:	5 31
Others	Bus state controller	
Package	144-pin plastic QFP (FP-144)	

The emulator operates in just the same way as the SH-2 on the user system and enables realtime emulation of the user system with functions for debugging hardware and software.

The emulator consists of a station and an emulator pod. The emulator pod should be connected directly to the user system.

1.2 Specification

The main features of the emulator are its emulation function, its floppy disk utility and its host system interface, as listed in tables 1.2 to 1.4.

Table 1.2 Emulation Functions

Command Type	Command	Function	Reference Section
Realtime emulation	GO	Performs realtime emulation in the following cases. The operating frequency is 28.5 MHz at max. • Executes until a hardware or PC break condition is satisfied, or until the (CTRL) + C keys or (BREAK) key is pressed. • Cycle-reset mode: Executes while the RES signal is sent to the MCU at fixed intervals. (Effective for waveform measurement immediately after a reset) • Parallel mode: Displays trace data and modifies memory contents during emulation.	7.2.19
	EXECUTION_ MODE	• Specifies execution mode.	7.2.17
Break condition setting	BREAK_ CONDITION 1,2	Sets hardware break conditions (1). • Normal break: Breaks when the following condition is satisfied (up to two points): — Address bus or data bus value — PC (program counter) value — Read/write condition — Delay count • Specification of the satisfaction sequence up to two points	7.2.6
	BREAK_ CONDITION 3,4	Sets hardware break conditions (2). • Address bus value (up to two points) • Read/write condition	7.2.6
	BREAK_ CONDITION 5	External probe trigger signal (one)	7.2.6
	BREAK	Sets software break conditions. • Normal break: Sets up to 255 breakpoints. • Sets pass count.	7.2.5

Table 1.2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Break condition setting (cont)	BREAK_ SEQUENCE	• Sequential software break up to four points and one reset point	7.2.7
Trace data acquisition and display	TRACE	• Displays executed instruction mnemonic. • Displays the following data for each bus cycle: — Instruction mnemonic — Address and data bus value — Access area and status — SH-2 I/O control signals — External probe value (one probe) — Clock count	7.2.40
	TRACE_ CONDITION	Sets trace condition. • Traces data only when a condition is satisfied. — Address bus value (NOT condition) — Read/write condition — Access type • Subroutine trace • Stops trace when a trace stop condition is satisfied. — Address and data bus value — Read/write condition — Access type — External probe value (one probe) — System control signals — NOT condition — Delay count • Outputs a low pulse from the trigger output pin when a condition is satisfied. — Address and data bus value — Read/write condition — Access type — External probe value (one probe) — System control signals — NOT condition — Delay count	7.2.41

Table 1.2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Trace data acquisition and display (cont)	TRACE_SEARCH	Searches for trace data.	7.2.44
	TRACE_MEMORY	Specifies memory address to trace.	7.2.42
Single-step execution	STEP	• Executes one step at a time, and displays the following. — Instruction mnemonic — Memory contents — Register contents • Displays the above data after a branch is executed • Executes displaying the above data for only specified routines. • This operation is performed for a specified number of steps or until a specified address is reached.	7.2.36
	STEP_ INFORMATION	• Specifies information to be displayed during single-step execution.	7.2.37
	STEP_OVER	• Executes subroutine as a single step.	7.2.38
Memory access	MEMORY, DUMP	Displays or modifies memory contents in 1-, 2-, or 4-byte units.	7.2.24, 7.2.15
	MAP	Specifies memory attributes in 128-kbyte units. • User memory • Write-protected area (128-kbyte units) • Guarded memory area (128-kbyte units) • Emulation memory 512 kbytes (SRAM with no wait state)	7.2.23
	FILL	Writes data in specified pattern.	7.2.18
	DATA_SEARCH, DATA_CHANGE	Searches for and replaces data in specified pattern.	7.2.13, 7.2.12
Clock selection	CLOCK	Selects emulator internal clock. • EML (6.144 MHz) Selects user system clock. • USER (28.5 MHz)	7.2.9

Table 1.2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Register access	REGISTER, .<register name>	Displays and modifies register contents.	7.2.31, 7.2.1
Line assembly	ASSEMBLE	Assembles instruction mnemonics and modifies memory contents. • Enables use of labels and symbol names.	7.2.4
Disassembly	DISASSEMBLE	Disassembles memory contents. • Displays labels and symbol names.	7.2.14
Execution time, pass count measurement	GO	Measures GO command execution time. • Measures total run-time (approx. 305 hours max).	7.2.19
	TRACE	Counts clocks in each bus cycle.	7.2.40
Test functions	FILL	Reads or writes the specified data to the memory.	7.2.18
	CHECK	Tests SH-2 I/O signals.	7.2.8
Symbolic debugging	LOAD, INTFC_LOAD	Loads symbols from host system.	9.4.2, 9.4.7
	SYMBOL, SHORT_SYMBOL	Defines symbols.	7.2.39, 7.2.34
	!<symbol name> &<symbol name>	Displays symbol contents according to attributes associated with symbol names. • Function names, label names • Variables (simple variables, pointer variables, arrays) and structure names • Line numbers	7.2.2
Command input	COMMAND_ CHAIN	• Provides automatic input from file. • Enables editing with cursor keys. • Copies immediately preceding line. • Copies operand of previous command.	7.2.10
	RADIX	Enables value input in binary, octal, decimal, hexadecimal, or ASCII characters. (Default can be specified)	7.2.30
Results display	PRINT	Outputs to printer or file.	7.2.28
	RESULT	Displays emulation results.	7.2.33

Table 1.2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Others	MOVE, MOVE_TO_RAM	Transfers memory contents. • Memory to memory • ROM (user system memory) to emulation memory	7.2.26, 7.2.27
	CONVERT	Converts number display. • Displays in binary, octal, decimal, hexadecimal, or ASCII characters.	7.2.11
	STATUS	Displays emulator operating status.	7.2.35
	GO	Monitors emulation. • Monitors emulation status every 200 ms and displays abnormalities found during emulation.	7.2.19
	RESET	Inputs RES signal to SH-2.	7.2.32
	MODE	Sets and displays the SH-2 operating mode.	7.2.25
	HELP	Displays all commands.	7.2.20
	HISTORY	Displays the history of the input command.	7.2.21

Table 1.3 Floppy Disk Utility Functions

Command Type	Command	Function	Reference Section
Backup	B (Monitor command)	Backs up and verifies floppy disk.	3.5.4 in Part I
File copy	FILE_COPY	Copies and verifies file contents.	8.4.1
Directory display	FILE_DIRECTORY	Displays file directory information of the floppy disk.	8.4.2
Dump	FILE_DUMP	• Dumps and modifies floppy disk. • Dumps and modifies file.	8.4.3
File deletion	FILE_ERASE	Deletes file.	8.4.4
Data transfer to and from user memory	FILE_LOAD	Loads file contents into memory.	8.4.5
	FILE_VERIFY	Verifies file contents against memory.	8.4.9
	FILE_SAVE	Saves memory contents in file.	8.4.7
Format	FLOPPY_FORMAT	Formats and initializes floppy disk.	8.4.11
File content display	FILE_TYPE	Displays file contents.	8.4.8
Rename	FILE_RENAME	Renames file.	8.4.6
Display free area	FLOPPY_CHECK	Displays free area on floppy disk.	8.4.10

Table 1.4 Host System Interface Functions

Command Type	Command	Function	Reference Section
Interface condition setting	HOST	Sets the following for RS-232C interface: • Transfer rate (300 to 19200 BPS) • Data length (7 or 8 bits) • Parity (even, odd, or none) • Number of stop bits (1 or 2 bits) • Busy control method (X-ON/X-OFF control or RTS/CTS control)	9.4.1
Program transfer to and from user system	LOAD, INTFC_LOAD	Loads contents from host system.	9.4.2, 9.4.7
	VERIFY, INTFC_VERIFY	Verifies file contents against memory.	9.4.6, 9.4.10
	SAVE, INTFC_SAVE	Saves memory contents in host system.	9.4.3, 9.4.8
Data transfer to and from floppy disk	TRANSFER, INTFC_TRANSFER	• Receives data from host system and writes it to file. • Transfers file contents to host system.	9.4.5, 9.4.9
Host system terminal selection	TERMINAL	Lets the host system operate as a terminal.	9.4.4

1.3 Realtime Emulation

The emulator enables realtime emulation for the SH-2 with no wait states. Realtime emulation consists of the following three modes:

- Normal mode
  Executes only emulation
- Cycle reset mode
  Forcibly inputs the RES signal to the SH-2 at a specified period
- Parallel mode
  Enables the user to display and modify memory and display trace information during user program execution

The user can select the mode which best suits his/her debugging needs. The following describes each of these modes.

1.3.1 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 and outputs a low pulse only once from the trigger output probe. When a number of times or sequential break for the software break condition is specified, the emulator stops the program execution every time the specified address is passed, and the emulator outputs a low pulse from the trigger output probe every time the software break condition is satisfied.

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

Gate Insertion Mode and Realtime Mode: The SH-2 microcomputer can access the user system memory in a single cycle at user system clock operation. This emulator, therefore, provides a gate insertion mode that gives priority to detailed debugging and a realtime mode that emphasizes realtime operation. Specify these modes with the EXECUTION_MODE command. For details on this command, refer to section 7.2.17, EXECUTION_MODE.

- Gate insertion mode
  The emulator controls access to user system memory and internal emulation memory. However, user memory and emulation memory cannot be located within the same area (CS*).

- Realtime mode

  This mode, which does not insert gates into the user bus, is provided to enable emulation without restricting access timing. Here, real time is given priority and the user bus and internal bus are connected directly. The user memory and emulation memory cannot be located within the same area (CS*).

Trace Information Acquisition during Cache Access: Acquiring trace information during cache accesses degrades the SH-2 operating speed (approximately 1.5 times) because the cache trace uses the external bus. Accordingly, to avoid degrading the SH-2 operating mode, disable the trace acquisition during cache accesses with the EXECUTION_MODE command. For details, refer to section 7.2.17, EXECUTION_MODE.

1.3.2 Cycle Reset Mode

Cycle Reset Mode Function: The emulator inputs the RES signal to the SH-2 at a specified period during realtime emulation and repeats the execution from the reset state. The emulator outputs a low-level pulse from the trigger output probe at the same time when the RES signal is input to the SH-2. This function is useful for observing waveforms from an initial state such as power-on reset to the specified time.

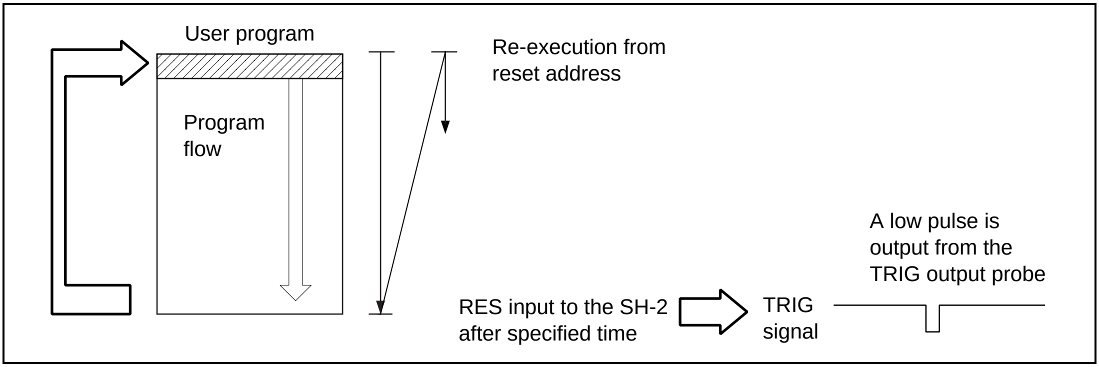

User program
Program flow
Re-execution from reset address
A low pulse is output from the TRIG output probe
RES input to the SH-2 after specified time
TRIG signal

Figure 1.1 Cycle Reset Mode

Cycle Reset Mode Specification: Set “R=n” as a GO command option to specify cycle reset mode.

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.

Note: When the emulator displays trace information after emulation terminates, disassembly of the instruction just before input of the RES signal may not be correct. If so, the mnemonic display will be .DATA.

Trigger Signal Output Timing in Cycle Reset Mode: In cycle reset mode, the emulator inputs the RES signal to the SH-2 when the time specified by a command passes.

Figure 1.2 shows the timing for output from the trigger output probe in cycle reset mode.

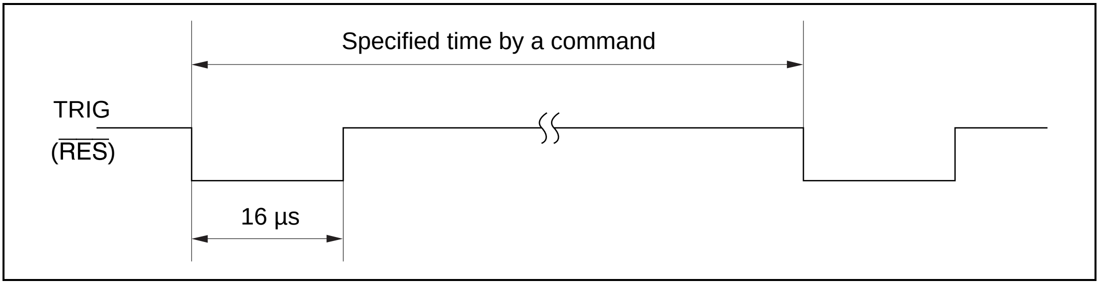


Figure 1.2 Trigger Signal Output Timing

1.3.3 Parallel Mode

Parallel Mode Function: In parallel mode, the emulator can display and modify memory or display trace information during realtime emulation.

Parallel Mode Specification: Parallel mode can be activated during GO command realtime emulation by any of the following methods as shown in figure 1.3.

- Press the (RET) key
- Press the space key
- Satisfy a trace stop condition specified by the TRACE_CONDITION 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.

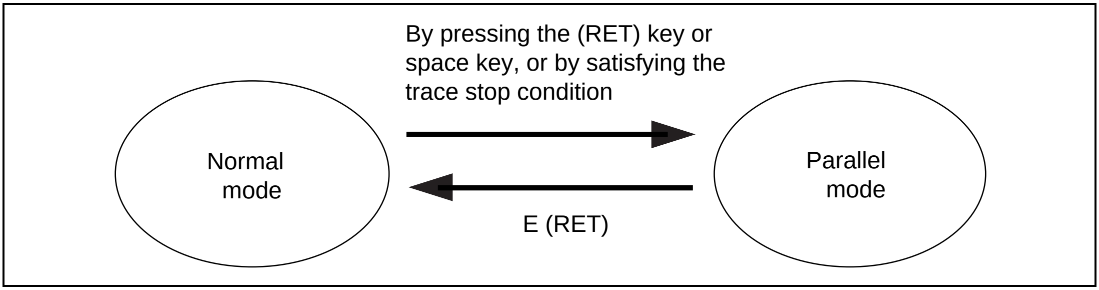


Figure 1.3 Transition to Parallel Mode

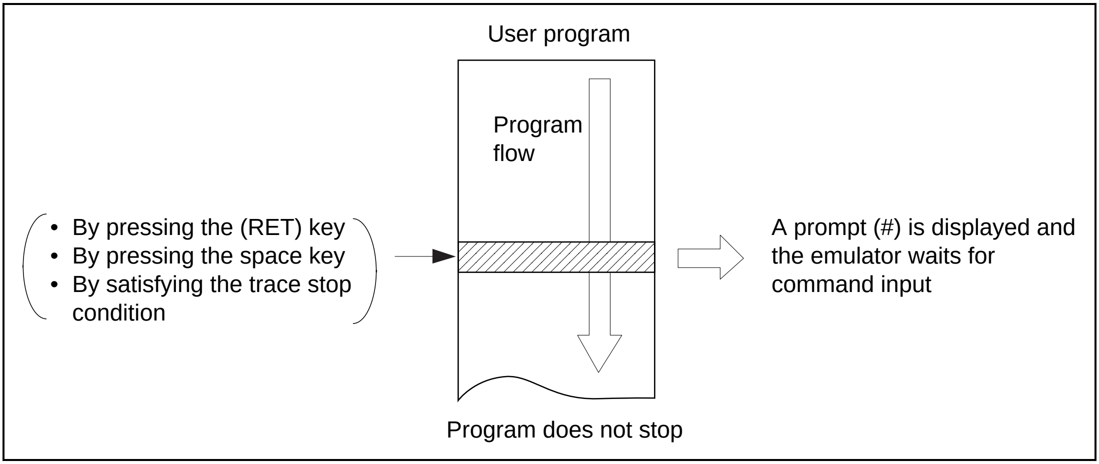

User program
Program flow
• By pressing the (RET) key
• By pressing the space key
• By satisfying the trace stop condition
A prompt (#) is displayed and the emulator waits for command input
Program does not stop

Figure 1.4 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 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_SEARCH, TRACE_CONDITION, or TRACE_MEMORY 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.

Table 12.1 lists the commands usable in parallel mode.

Notes: 1. When memory (emulation memory, internal I/O, and user system memory) is accessed with the MEMORY command, DUMP command, DISASSEMBLE command or !<symbol name> 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 0.x ms (undefined) during user system clock operation. Therefore, realtime emulation cannot be performed.

— Internal I/O and user memory
When accessing internal I/O and user memory, the user program temporarily halts. This pause lasts for about 0.x ms during user system clock operation.

— 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
- !<symbol>: During symbol read

2. During execution of the TRACE, TRACE_SEARCH, TRACE_MEMORY or TRACE_CONDITION command, the emulator stops trace information acquisition.

3. The emulator cannot enter parallel mode when executing emulation in the following modes specified by GO command mode options.

- Cycle reset mode

1.4 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 protect/guarded break
  Caused by writing to a write-protected area or accessing guarded area

1.4.1 Hardware Break

Break conditions can be specified at any five points with the BREAK_CONDITION1 to 5 commands. Only address bus value and read/write conditions (AND condition) can be specified with the BREAK_CONDITION3,4 commands, and only external probe value conditions for multibreak can be specified with the BREAK_CONDITION5 command.

(1) Break Conditions

BREAK_CONDITION1, 2:

- Address bus value
- PC (program counter) value (value before or after execution can be selected)
- Data bus value (BREAK_CONDITION1 only)
- Read/write condition
- Delay count (H'1 to H'FFF bus cycles)
- Sequential break
- Access type (DAT, DMA)

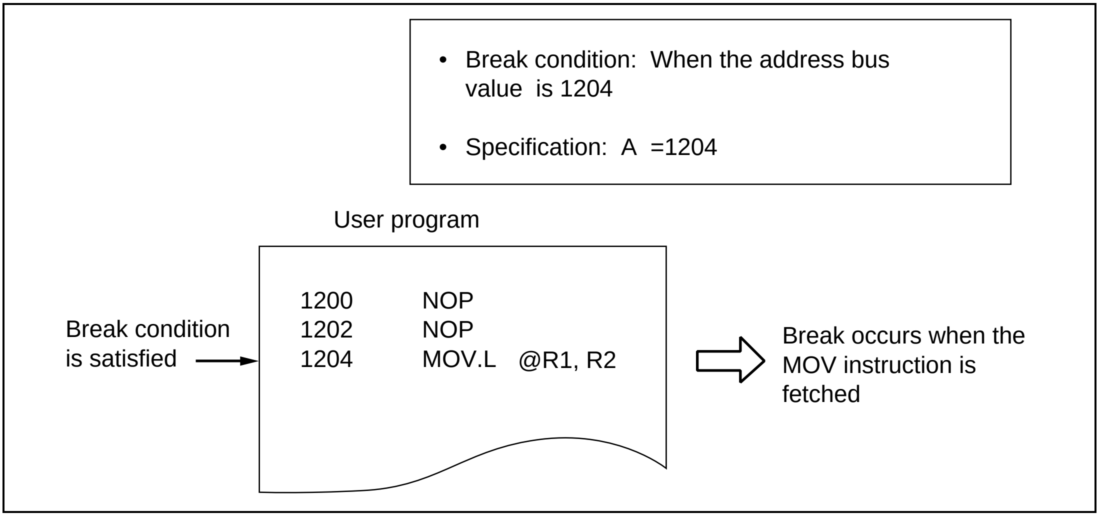
BREAK_CONDITION3,4:

- Address bus value
- Read/write condition (ANDed with the address bus value condition)

BREAK_CONDITION5:
Falling-edge signals from the external probe pin

Note: Because the SH-2 prefetches instructions, a break may occur before the prefetched instruction is executed when a break condition is satisfied during a prefetch cycle.

Address Bus Value: Address bus values can be specified with the BREAK_ CONDITION1,2,3,4 commands. A break occurs when the SH-2 address bus value matches the specified condition. To specify a break address, there are two methods. One is to specify an address range, and the other is to specify a particular address.



• 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 1.5 Break with Address Bus Value

Data Bus Value: Data bus values can be specified with the BREAK_CONDITION1 command. A break occurs when the SH-2 data bus value matches the specified condition. The emulator checks both program fetch and data access for the condition.

When specifying the condition, the access size must be selected from long word access (LD), word access (WD), or byte access (D).

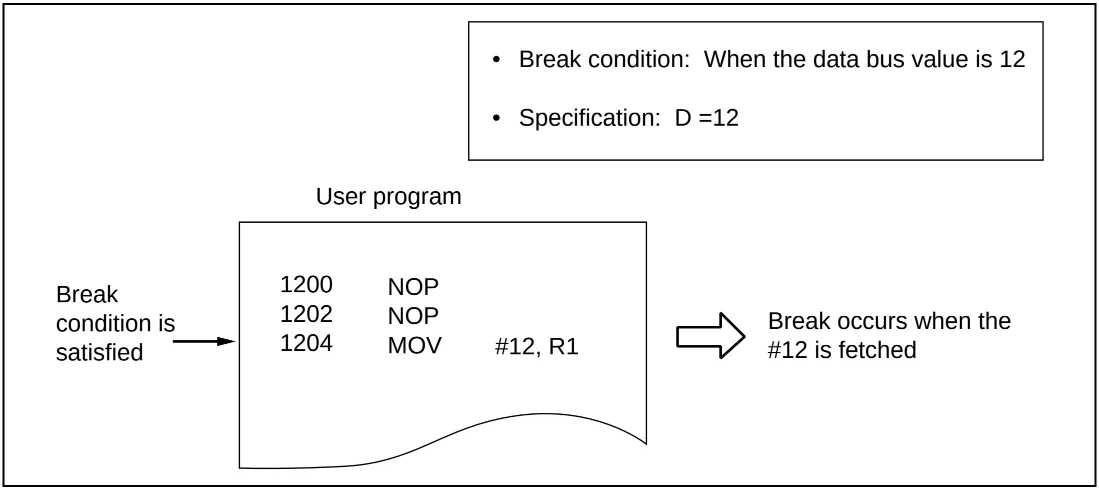

• Break condition: When the data bus value is 12
• Specification: D =12
User program
1200 NOP
1202 NOP
1204 MOV #12, R1
Break condition is satisfied
Break occurs when the #12 is fetched

Figure 1.6 Break with Data Bus Value

Read/Write Condition: Read/write condition can be specified with the BREAK_CONDITION1,2. A break occurs when the SH-2's RD and RDWR signal levels match the specified conditions. Usually, the read/write condition is specified together with the address or data conditions.

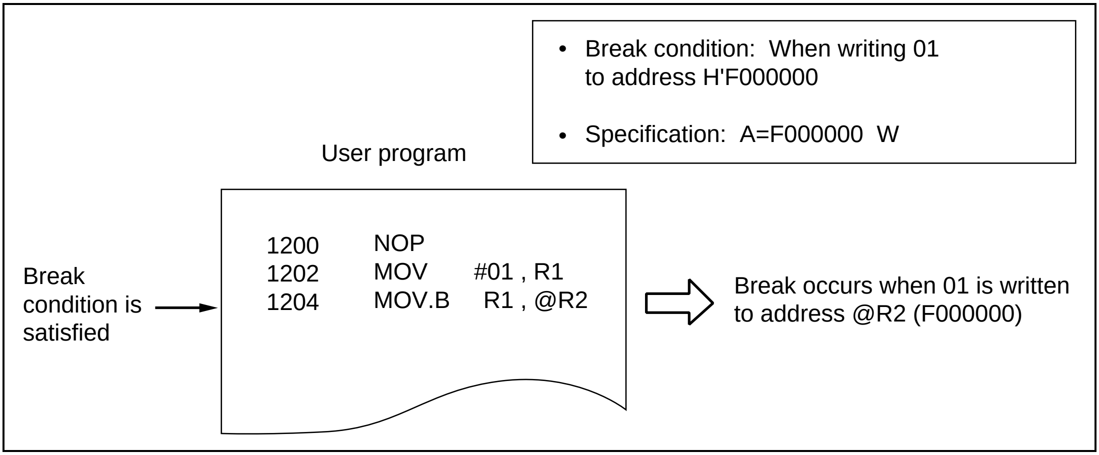
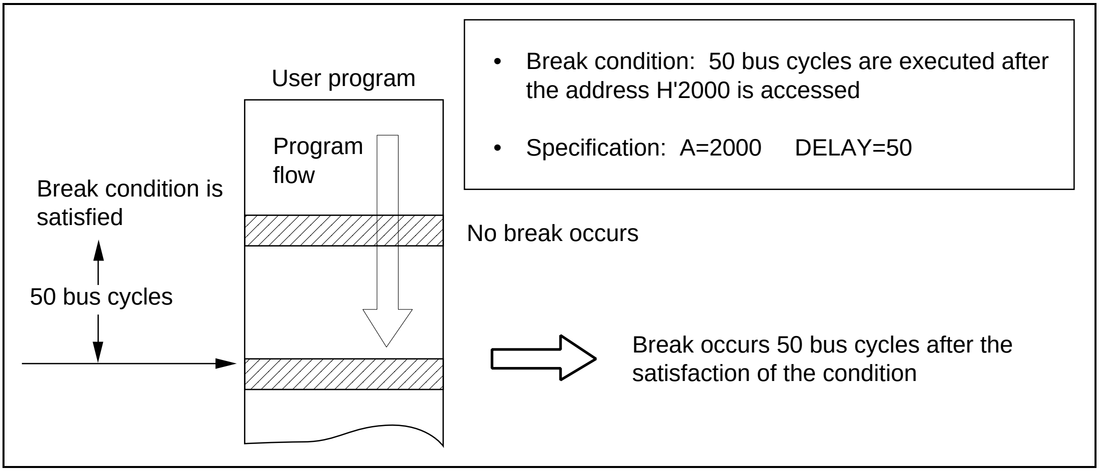

• Break condition: When writing 01 to address H'F000000
• Specification: A=F000000 W
User program
1200 NOP
1202 MOV #01 , R1
1204 MOV.B R1 , @R2
Break condition is satisfied
Break occurs when 01 is written to address @R2 (F000000)

Figure 1.7 Break with Read/Write

Delay Count: A break occurs when the above break condition is satisfied and the emulator executes the bus cycle for a specified number of times (4,095 max). When specifying this condition, specify in combination with any of the above break conditions. The user can specify this function only with the BREAK_CONDITION1 command.


User program
Program flow
Break condition is satisfied
50 bus cycles
No break occurs
Break occurs 50 bus cycles after the satisfaction of the condition

- Break condition: 50 bus cycles are executed after the address H'2000 is accessed
- Specification: A=2000 DELAY=50

Figure 1.8 Break with Delay Count Specification

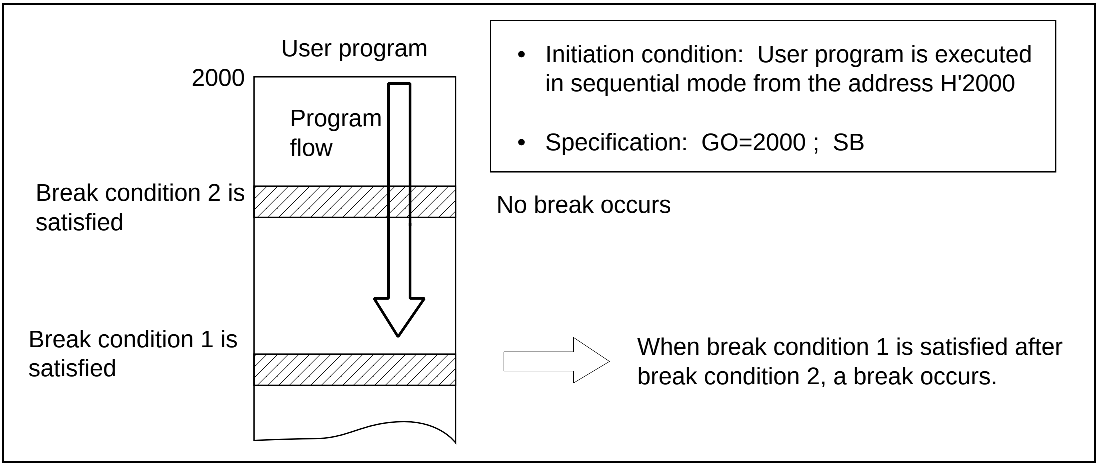
Sequential Break Condition: In sequential mode, a break occurs when hardware break conditions 2 and 1 have been satisfied in that order.

- Sequential break mode

  When break condition 2 and then break condition 1 are satisfied, a break occurs.

Specify the break condition with the BREAK_CONDITION1,2 commands. The user can specify any of the above conditions.

When executing the user program, specify the sequential break option (;SB) with the GO command. When no option is specified, the sequential break function does not operate, and a break will occur as soon as any of the break conditions are satisfied.


User program
2000
Program flow
Break condition 2 is satisfied
Break condition 1 is satisfied
No break occurs
When break condition 1 is satisfied after break condition 2, a break occurs.

- Initiation condition: User program is executed in sequential mode from the address H'2000
- Specification: GO=2000 ; SB

Figure 1.9 Break with Sequential Specification

External Probe Value: External probe value can only be specified with the BREAK_CONDITION5 command.

- A break occurs when a falling edge of the external probe signal is detected.
- Multibreak function

  The external probe has a falling-edge detection function. By using this function, you can generate a break in other emulators of the same type simultaneously. The following explains how to perform a multibreak with the external probe.

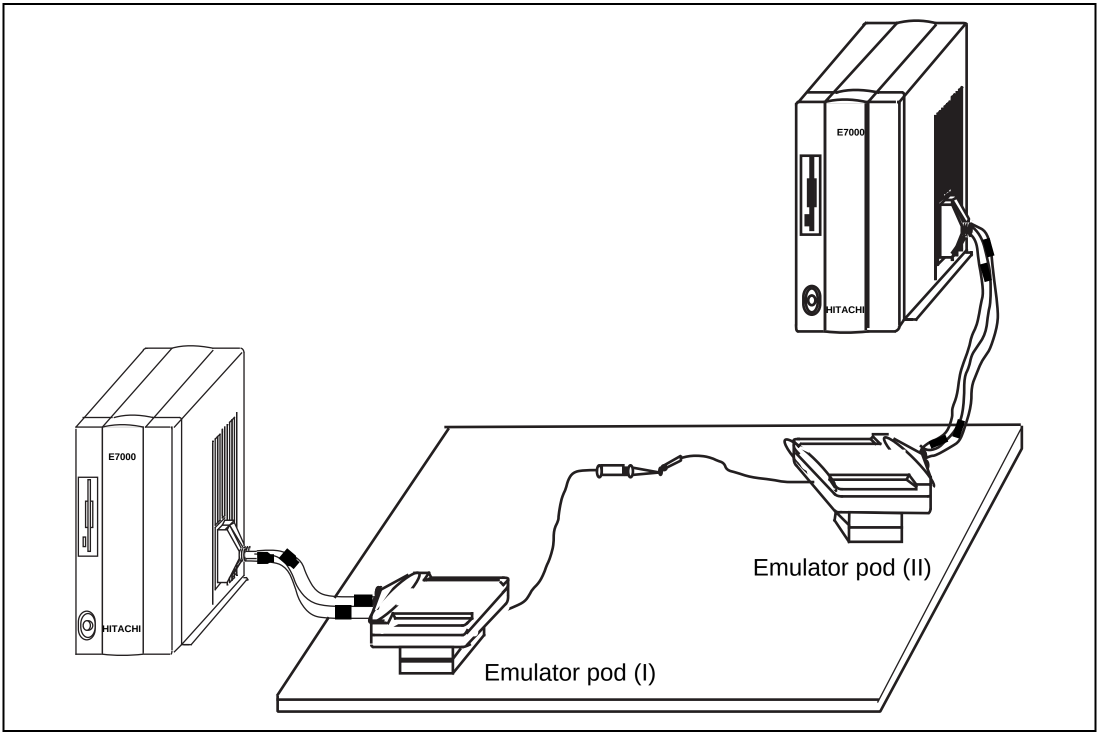


Figure 1.10 Multibreak Function

Procedure

1. Connect the external probe of emulation pod (I) to the trigger output probe or the RUN/break status signal of emulator pod (II).

2. Set the break condition for emulator pod (II) and specify trigger output mode by the TRG option of the EXECUTION_MODE command. For details, refer to section 7.2.17, EXECUTION_MODE command.

3. Set the break condition for the external probe of emulator pod (I) by the PRB condition of the BREAK_CONDITION5 command. For details, refer to section 7.2.6, BREAK_CONDITION5 command.

4. Execute the programs for both emulator pods (I) and (II).

5. First, when the break condition is satisfied, the emulator pod (II) program breaks. At this time, the emulator pod (I) program also stops due to pulse output from the trigger output probe or the RUN/break signal. However, the break in emulator pod (I) follows that in emulator pod (II) by several bus cycles.

In the multibreak function, the following points must also be noted.

- When the RUN/break status signal (pin 5) is used as an output

  A multibreak occurs if the program stops when the condition specified by BREAK or the BREAK_CONDITION1,2 command is satisfied, the BREAK key is pressed, or a write-protected area is accessed. Note, however, that the following must not be performed to prevent the undesignated E7000 from breaking.

  — Performing parallel access
  — Specifying a number of passes as a condition of the BREAK command
  — Executing BREAK_SEQUENCE command
  — Setting software breakpoints at the start address of the GO command
  — Setting the PC breakpoint at the start address of the GO command with the BREAK_CONDITION1,2 command

- When the trigger output probe pin (pin 2) is used as an output

  A multibreak occurs only at the breakpoint specified with the BREAK_CONDITION1,2 command. In this case, TRG = E must be specified as input with the EXECUTION_MODE command. Note, however, that the following must not be performed to prevent the undesignated E7000 from breaking.

  — Setting cycle reset mode with the GO command
  — Setting trigger mode with the BREAK_CONDITION1,2 command
  — Specifying a delay condition with the BREAK_CONDITION1 command
  — Specifying trigger mode with the TRACE_CONDITIONcommand

(2) Break Timing

Hardware break sampling timing synchronizes with the SH-2 bus cycle (DS signal).

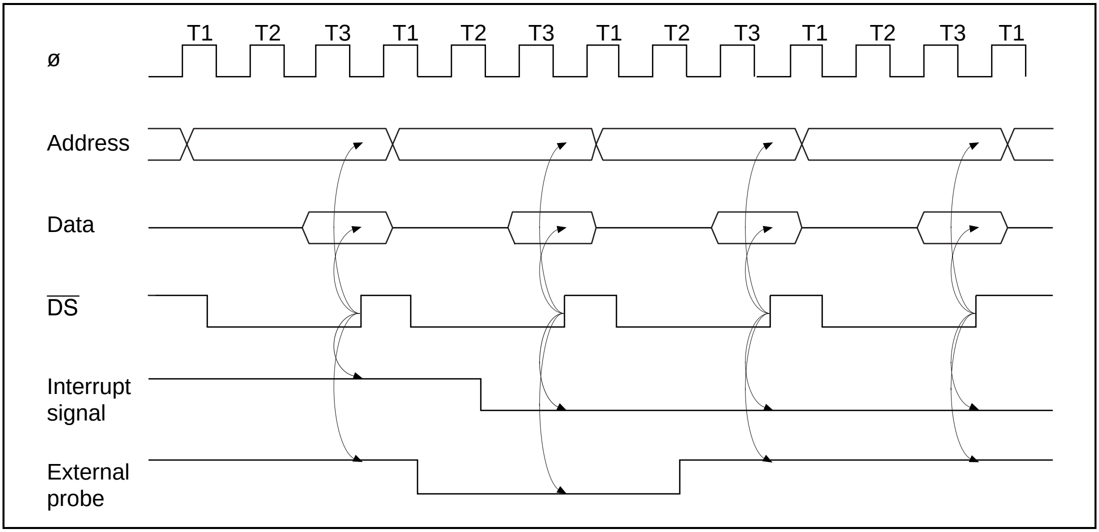


Figure 1.11 Break Timing

Notes: 1. Because the SH-2 prefetches instructions, a break may occur before the prefetched instruction is executed or after multiple instructions are executed when a break condition is satisfied during a prefetch cycle.

2. When the DS signal is negated simultaneously with the external probe signal transition, the break condition may be undefined while a value on that external probe is specified as a break condition.

3. The above DS signal, which is an emulator internal signal, represents the bus cycle. (Hereinafter, the DS signal is used to indicate bus cycles.)

1.4.2 Software Break

The contents at the specified address is replaced with a break instruction. The program execution stops when the break instruction is executed. The replaced instruction at the address is not executed. 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 breaks

Normal Break:

A break occurs after executing the breakpoint instruction 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 breakpoint instruction a specified number of times. The maximum number to specify is 16383 (H'3FFF).

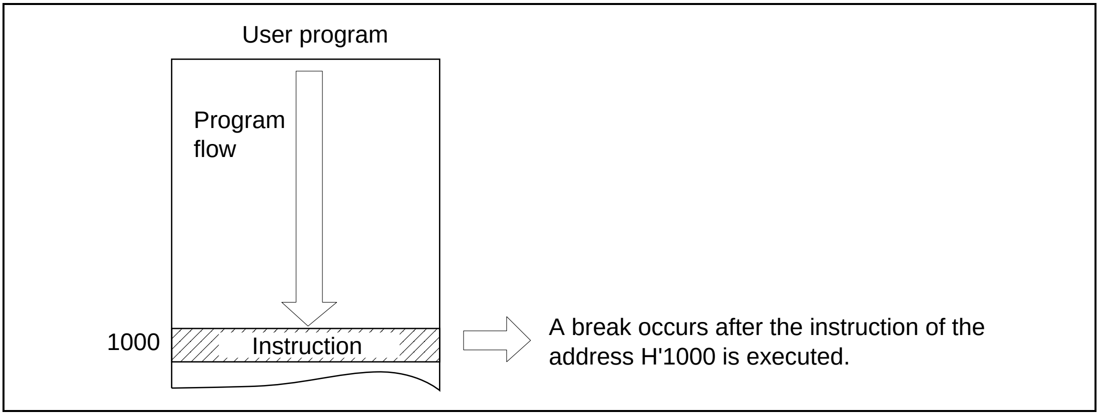

User program
Program flow
1000
Instruction
A break occurs after the instruction of the address H'1000 is executed.

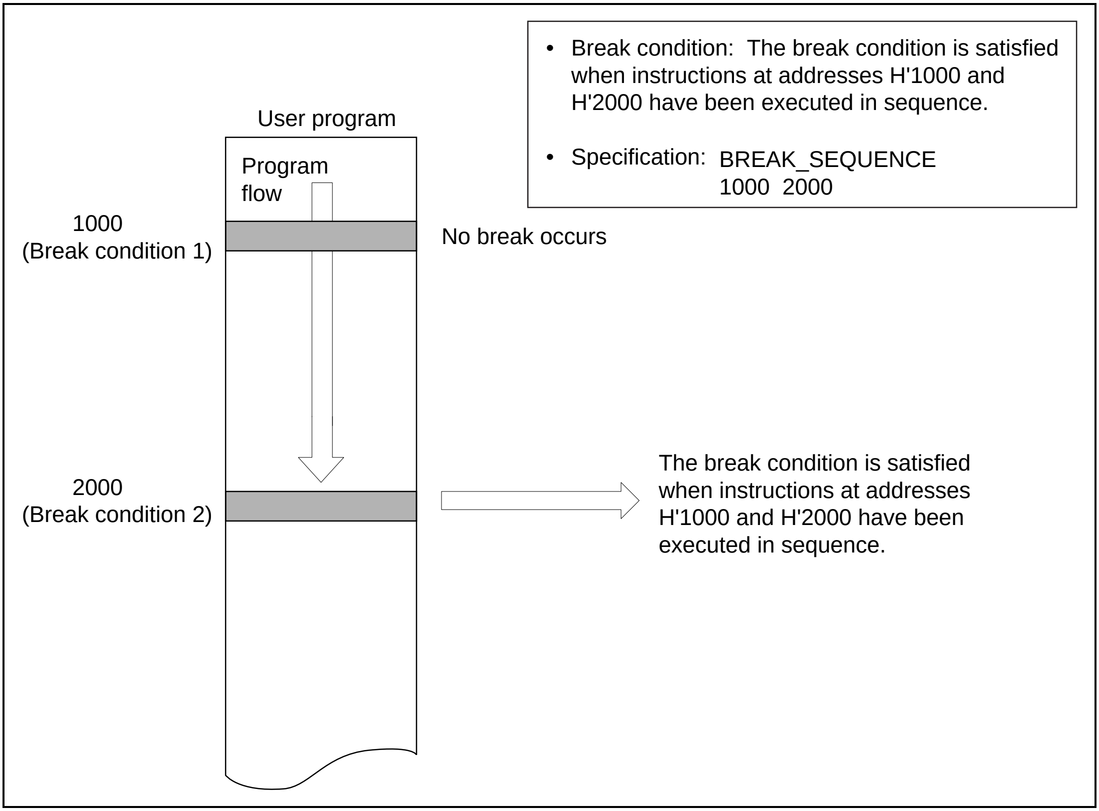
Figure 1.12 Normal Break (Software Break)

Note: When specifying the number of times that the break condition is to be satisfied before generating a 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 program execution. At this time, the BC2 condition becomes invalid because the BC2 condition is used to perform the step execution of the break address.

Sequential Break:

A sequential break occurs (two points max) when certain conditions are satisfied in a specified order. A reset point can be specified in addition to these two points. If the reset point is reached, all sequential break conditions up to that point become invalid and the emulator rechecks from the first break condition.

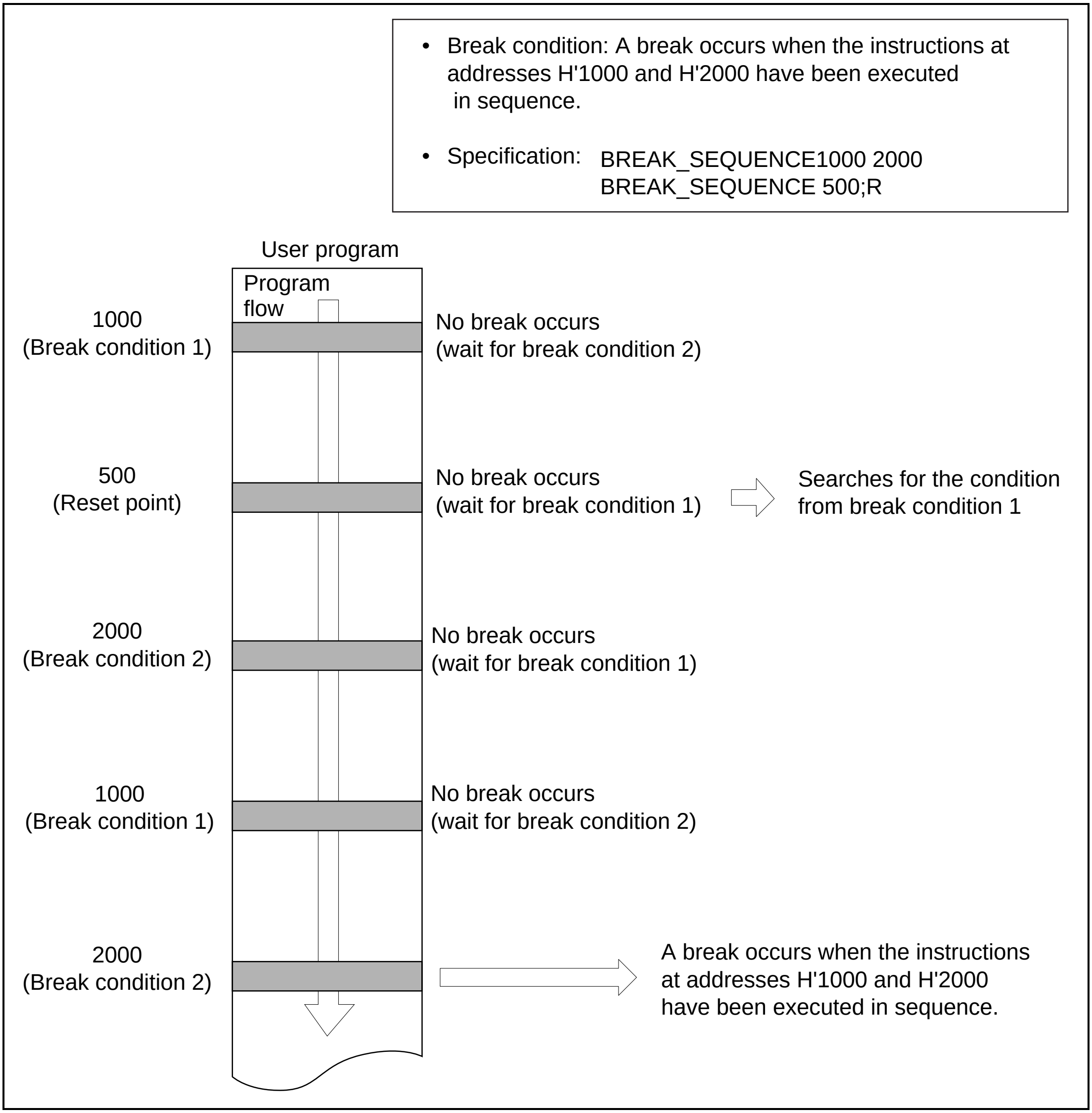
Figure 1.13 illustrates the usual sequential break and figure 1.14 describes a sequential break when a reset point is specified.



- 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

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 1.13 Sequential Break



- Break condition: A break occurs when the instructions at addresses H'1000 and H'2000 have been executed in sequence.
- Specification:

```
BREAK_SEQUENCE1000 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 1.14 Sequential Break (Reset Point Specification)

Note: When specifying the sequential break, 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. Accordingly, the BC2 settings are invalid at pass point or reset point execution.

1.4.3 Forced Break

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

1.4.4 Write Protect/Guarded Break

The user can specify the SH-2 memory area as write-protected or guarded areas in 128-kbyte units with the MAP command. The emulator forcibly stops the program when a write access or read/write access is attempted to the specified memory. A normal break occurs, however, after the accessed instruction execution terminates. For details, refer to section 7.2.23, MAP.

1.5 Realtime Trace Function

The emulator can trace bus information during realtime emulation without affecting the user system. The emulator can fetch bus information of the SH-2 address or data, and the external probe value up to 32, 767 bus cycles. Trace information is displayed with the TRACE command. Display of this information enables a check on executed program.

Trace information:

- Address bus: 26 bits (PC value: 32 bits)
- Data bus: 32 bits
- External probe: 1
- SH-2 I/O control signals: 30
- Number of bus cycle clocks (ø): 8 bits (128 kbytes max)
- Memory contents tracing: 32 bits (internal 32 bits)

Emulator displays trace information as the following methods:

- Fetches only instruction words from the trace information and displays them in mnemonic.
- Displays the trace information as mnemonic in bus cycle units.
- Searches for the specified information and displays it. Use the TRACE_SEARCH command.

1.5.1 Trace Timing

Trace information is acquired in trace memory synchronized with rising edges of the data strobe (DS) signal. However, because external probe signal input is not synchronized with the DS signal, it may not be possible to log all the changes in the external probe signal.

In each bus cycle, the clock number is the number of clock (ø) cycles between the end of the previous bus cycle and the end of the current bus cycle (between one rising edge of DS and the next). Figure 1.15 shows an example of the external probe trace signal.

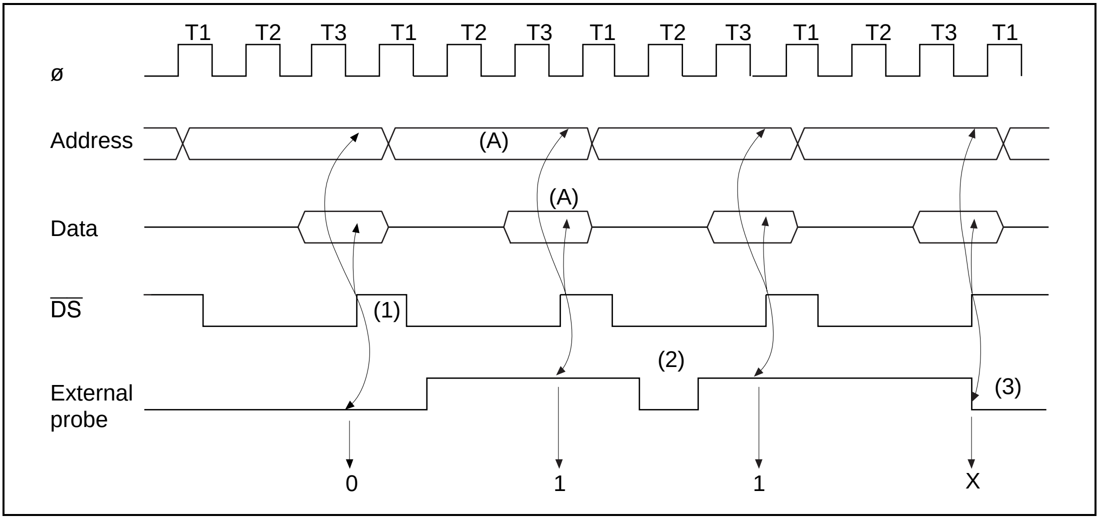


Figure 1.15 External Probe Trace Signal

Example:

- External probe signal
  - — Trace information sampled at rising edges of DS (figure 1.15 (1)).
  - — When the external probe signal changes between samplings, it cannot be reflected in the trace data (figure 1.15 (2)).
  - — When a sampling edge coincides with a change in the external probe signal, the trace contents are unfixed (figure 1.15 (3)).
- Clock number
  - — Three clock cycles are traced in bus cycle (A).

1.5.2 Trace Condition Setting

The user can specify the following four conditions with the TRACE_CONDITION command.

- Free trace
- Subroutine trace
- Range trace
- Trace stop

Free Trace: In free trace when the user program is executed as a result of the GO, STEP, STEP_OVER command, tracing is carried out continuously for a maximum of the latest 32,767 bus cycles until a break condition is satisfied. When no parameter is given with the

TRACE_CONDITION command, the default is free trace. Figure 1.16 illustrates the free trace operation.

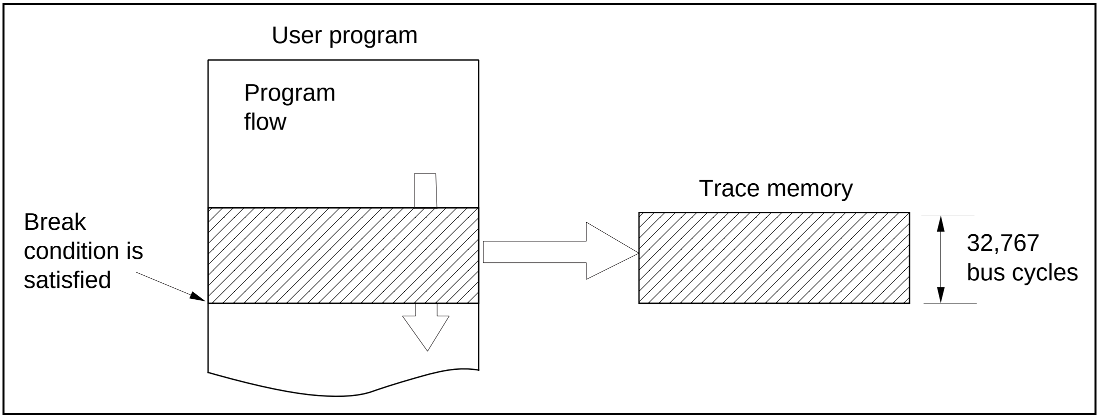


Figure 1.16 Free Trace Execution

Subroutine Trace: When a subroutine trace is specified, the emulator acquires operand accesses and instructions between a specified start address and end address. However, when the specified subroutine calls another subroutine, the called subroutine is not traced. Figure 1.17 illustrates the operation of the subroutine trace.

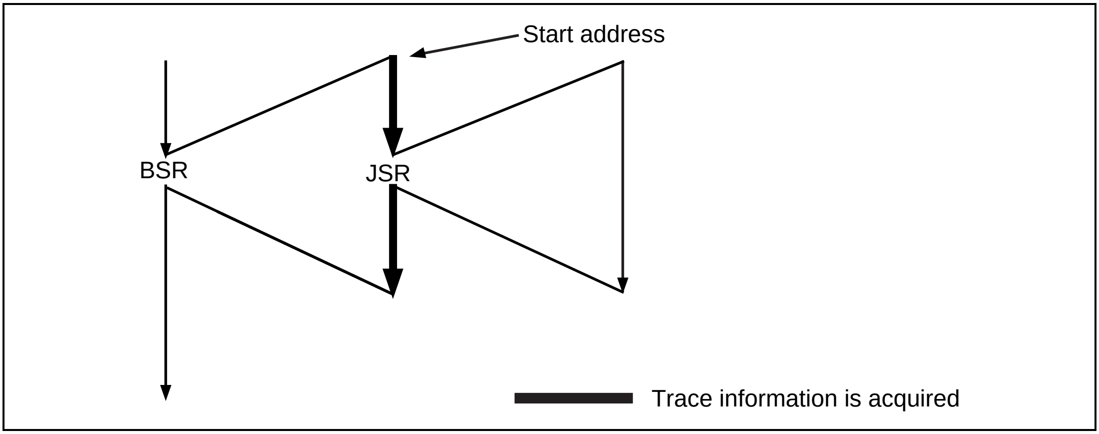


Figure 1.17 Subroutine Trace Specification

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 value (Within or outside a specified range)
- Read/write condition
- Access type

Figure 1.18 illustrates the trace acquisition condition.

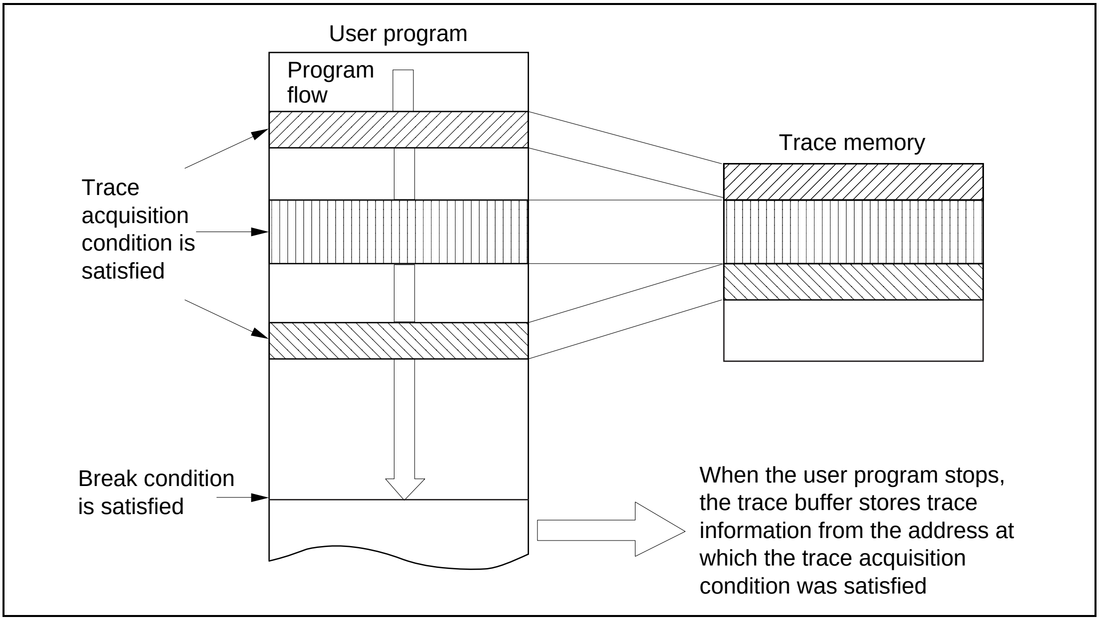

User program
Program flow
Trace memory
Trace acquisition condition is satisfied
Break condition is satisfied
When the user program stops, the trace buffer stores trace information from the address at which the trace acquisition condition was satisfied

Figure 1.18 Trace Acquisition Condition

Trace Stop (Parallel Mode): When a trace stop condition is specified, the emulator acquires trace information until the specified condition is satisfied. At this point, trace acquisition stops and the emulator prompts for command input, although realtime emulation does not stop. This so-called parallel mode enables the user to view trace information without stopping realtime emulation. 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 and data bus value
- Read/write condition
- Access type (PRG, DAT, DMA)
- External probe value
- System control signal (BREQ)
- NOT condition
- Delay count (1 to H'7FFF)

The trace stop condition is set with the TRACE_CONDITION command. Figure 1.19 illustrates how the system operates when a trace stop condition is specified.

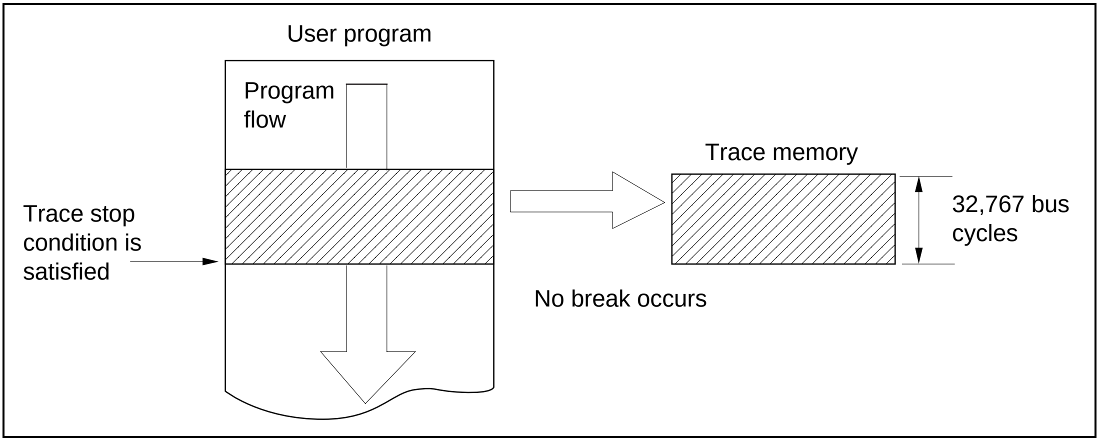


Figure 1.19 Trace Stop Condition Specification

1.5.3 Trace Display

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

- Instruction Display

  From the contents of the trace information, only the executed instructions are displayed in mnemonics.

- Bus Cycle Display

  Trace information is displayed in bus cycle units. When a trace is specified in special address memory, the memory contents are also displayed. The memory address is specified with the TRACE or MEMORY command.

- Retrieval Display

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

1.6 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.

- SH-2 control registers (PC, SR, PR, GBR, VBR, MACH, MACL)
- SH-2 general-purpose registers (R0 to R15)
- Instruction address
- Instruction mnemonic
- Memory contents
- The reason for stopping

1.6.1 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 a single 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, BTS, BFS, BRAF, BSRF, TRAPA

- Subroutine display

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

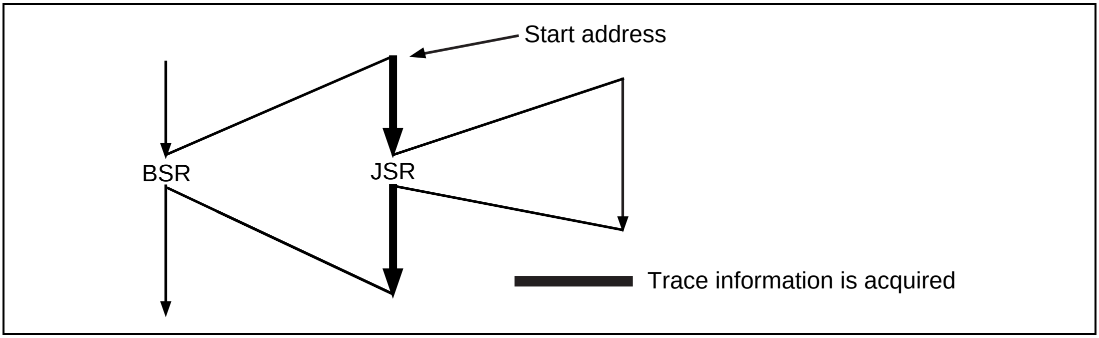


Figure 1.20 Subroutine Display

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

- Subroutine step execution:

  When executing a BSR, JSR, or TRAPA instruction, the emulator treats the called subroutine as a single step. All other instructions are executed one at a time. The subroutines may be located in the user RAM or the emulation memory area.

1.6.2 Setting Display Information

The user can set the information displayed at each instruction using the STEP_INFORMATION command.

1.6.3 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 continuous execution by specifying a stop address. However, the specified address must be at the start of an instruction.

1.7 Trigger Output

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

- Cycle reset
- Trace condition satisfaction (When trigger output is specified)
- Hardware break condition satisfaction

When using this pulse as an oscilloscope trigger input signal, it becomes easy to adjust the user system hardware.

Cycle Reset: The emulator outputs a low-level pulse from the trigger output probe with the same timing as the RES input signal to the SH-2. Cycle reset is specified with the GO command. For cycle reset timing, refer to section 1.3.2, Cycle Reset Mode.

Trace Condition Satisfaction: When the trigger output is specified using the TRACE_CONDITION command, a low-level pulse is output from the trigger output probe at bus cycles corresponding to the specified condition. The trigger signal is output from the end of the corresponding bus cycle until the end of the next bus cycle. If the conditions are satisfied in consecutive bus cycles, the trigger output remains low.

Hardware Break Condition Satisfaction: During emulation, a low-level pulse is output from the trigger output pin at the end of the bus cycle during which the hardware break condition is satisfied. The trigger signal is output from the end of the corresponding bus cycle until the end of the next bus cycle. If the conditions are satisfied in consecutive bus cycles, the trigger output remains low.

Figure 1.21 shows the output timing.

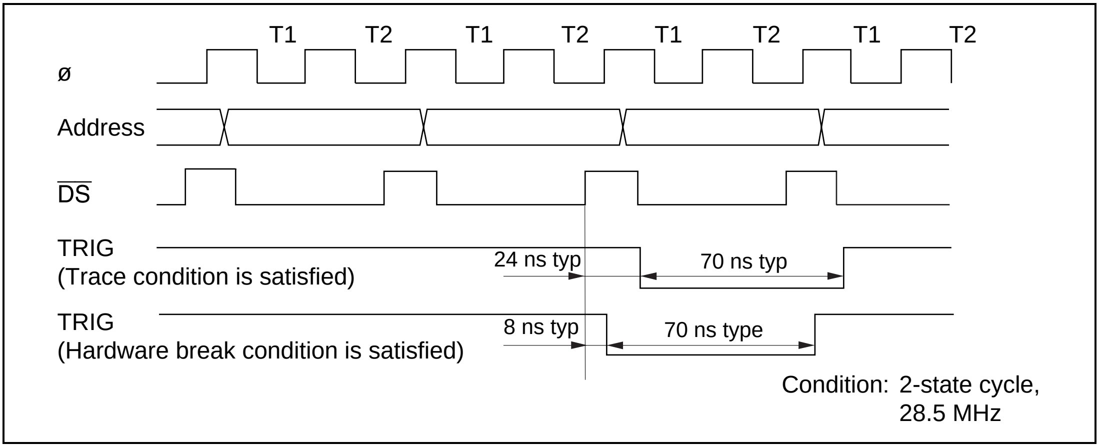


Figure 1.21 Pulse Output Timing

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.

1.8 Memory Access Function

Memory Management: The SH-2 has a maximum address space of 128 Mbytes × 4 areas. The emulator manages this space in units of 128 kbytes with the following memory attributes. Note that user memory and emulation memory cannot be located within the same area.

- User memory (User system memory)
- Emulation memory (512-kbyte standard memory)
- User memory write protect
- Emulation memory write protect
- User memory access-prohibited (Guard Function)
- Emulation memory access-prohibited (Guard Function)

For details, refer to section 3.2, Memory Space, and section 7.2.23, MAP.

Memory Display, Modification, and Transfer: Memory display, modification and transfer are done, using the following command.

ASSEMBLE	DATA_CHANGE	DUMP
FILL	MEMORY	MOVE
MOVE_TO_RAM		
LOAD	SAVE	
FILE_LOAD	FILE_SAVE	
LAN_LOAD	LAN_SAVE	
INTFC_LOAD	INTFC_SAVE	

1.9 SH-2 Control and Status Check

The emulator is capable of switching the clock signal supplied to the SH-2, checking normal operation, 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) and the internal clock (6.144 MHz). To switch the clock, refer to section 7.2.9, CLOCK, and note the followings.

- When the clock is switched, the emulator inputs a RES signal to the SH-2. 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. In this case, if clock mode 4 to 6 is specified, an error message is displayed but the internal clock is not selected.

- When initiating the emulator system program, the emulator selects the SH-2 clock automatically in the following order.
  - When an external clock is supplied from the user system, selects the user system clock
  - Selects the internal clock (6.144 MHz)

Strobe Signal Output Function at Emulation Memory Access: When the realtime mode or gate insertion mode is specified with the EXECUTION_MODE command and emulation memory is accessed, the emulator outputs the corresponding address to the user system, but does not output the strobe signals (CS0 to CS3, RD, RDWR, CKE, RAS, and CAS signals).

Check of the I/O signals: The emulator checks the connection of the user system at the system initiation. By this check, abnormalities such as short of the user system interface signal can be detected. The signals for the check is as follows.

The CHECK command can check the same signals with the ones at the system initiation.

RES, BREQ, WAIT, IRL0 to IRL3, NMI

Emulator Execution Status Display: The emulator can display the execution status information listed in table 1.5. To display the execution status, use the STATUS command.

Table 1.5 Execution Status Display

Operation Status
MCU type
SH-2 operating mode
Radix type
Number of breakpoints specified with the BREAK command
BREAK_CONDITION 1 to 5 command specification
TRACE_CONDITION command setting status
Host system interface conditions
Number of defined symbols
Number of defined line number symbols
Information displayed with the STEP_INFORMATION command
Address range displayed with the STEP command
Type of clock specified by the CLOCK command
File name or printer that outputs the CRT displayed contents
Extension function display
Remaining emulation memory

1.10 Emulation Monitoring Function

The SH-2 emulator monitors the emulation status such as memory accesses or user program execution. Two kinds of status are monitored.

- SH-2 operation status
- User system power and clock status

SH-2 Operating Status: When executing the program with the GO command, the emulator monitors the execution status every 200 ms. When the status changes, the execution status display is updated. With this function, the user can observe the progress of the program. Next, the execution status is displayed. The execution status cannot be output to a printer or file assigned with the PRINT command. For details, refer to the description on execution status display, in section 7.2.19, GO.

Table 1.6 Operating Status Display

Display	Meaning
** RUNNING*[1]	The user program execution is initiated. This message is displayed once when GO command execution is started or when parallel mode is cancelled. Note that this message will be deleted when **PC=xxxxxxxx [yyyyyy=zz] is displayed.
**PC=xxxxxxxx [xxxxxxxx = xxxxxxxx] (a) (b) (c)	
(a) Program fetch address (b) Memory address (c) Memory contents	Current program fetch address is displayed. When specifying TM option with GO command, the memory address and its contents are displayed.
** VCC DOWN	User system Vcc (power voltage) is 4 V or less.
** RESET	RES signal is low. The SH-2 has been reset.
** WAIT A = xxxxxxxx xxxxxxxx: Address bus value	WAIT signal is low.
** BREQ*[2]	BREQ signal is low.
** TOUT A = xxxxxxxx*[2] xxxxxxxx: Address bus value	The address bus value is displayed. The bus cycle stops for 80 μs or more.
** PREQ	Clock pause signal is low.

Notes: 1. **PC=xxxxxxxx [yyyyyy=zz] will not be displayed if the cache access trace is disabled by specifying D in the CT option of the EXECUTION_MODE command or if the user program is executed only in the cache.

2. These messages are not displayed if the cache access trace is disabled.

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 SH-2 clock is set to “USER” with the CLOCK command, the emulator executes the following operation according to the emulator status.

- During user program execution
  - When the user system is turned off (Vcc is 4V 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 is halted, USER SYSTEM NOT READY (NO CLOCK) is displayed and the emulator system program stops. To operate the emulator again, restart the system program.

- During command input wait state
  - — When the user system is turned off (Vcc is 4 V or lower), USER SYSTEM NOT READY (NO CLOCK) is displayed and the SH-2 operating clock is switched to the internal 6.144-MHz clock and the emulator waits for command input. A RES signal is input to the SH-2, and the internal registers are initialized. USER SYSTEM NOT READY (NO CLOCK) is displayed after the user system has been turned off and one command has been executed.
  - — When the clock stops (Vcc is over 4 V), USER SYSTEM NOT READY (NO CLOCK) is displayed and the emulator system terminates. Restart the emulator in order to continue emulation.

Notes: 1. If the user system power is turned off, this is detected (Vcc is 4 V or lower) before the clock termination is detected.

2. Clock halt means that only the clock halts and the user system remains on.

1.11 Symbolic Debugging

The emulator has a symbolic debugging function which uses the source program symbols (variables and line numbers). This function is explained below.

1.11.1 Defining Symbols

- Definition Methods

  There are three ways to define symbols.

  - — Load module with debugging information load

    A linkage editor can be used to create a load module containing information for debugging (SYSROF-type load module). Symbols can be defined by loading such a module with the LOAD command.

  - — SYMBOL command

    Symbols can be defined with the SYMBOL command. In this case, the symbol attributes are registered as label.

  - — ASSEMBLE command

    Label names are defined using the ASSEMBLE command.

- Symbol Attributes

  Table 1.7 lists the attributes of the symbols which may be defined within the load module (SYSROF type) debugging information.

Table 1.7 List of Symbol Attributes

Item	Attributes
Symbol type	Function name
	Structure name
	Label name
	Variable name (Simple variable, pointer variable, array)
	Line number
Allocation type	External static variable
	Internal static variable
Variable type	Character
	Integer
	Pointer

- Symbol Names and Abbreviations

  Symbols within load module debugging information are nested. Use a slash (/) to separate the different parts of a symbol in this nested structure. The basic symbol format is shown below.

  — Basic symbol format

  !<unit name>/<function name>/<variable name>

  — Line number symbol

  &<unit name>/<line number>

To simplify the nested structure, abbreviations can be defined for symbol names up to a specific level in a nested structure.

1.11.2 Symbol Reference

Using Symbols in Commands: A symbol can be used as an address or data in a command after it has been defined by the SYMBOL command. In this case, the value of the symbol is treated as the input value.

Referring to Symbol Contents: If a symbol is input in the following format in command input wait state, the symbol's contents are displayed:

```
:!<symbol name>
```

If the D option is specified, it will be displayed in decimal.

```
:!<symbol name>  D           (D: Option for symbol value in decimal)
```

1.11.3 Symbol Deletion

The symbol deletion function of the SYMBOL command deletes all symbols at the same time. Specific symbols cannot be deleted.

1.11.4 Symbol Display

Displaying Defined Symbols: The SYMBOL command can be used to display either all the symbols or a specific symbol. This command displays the symbol name, its address, and its attribute.

During Disassembly: When there are symbols that correspond to labels and operands of absolute addressing modes (PC relative, absolute address), the symbols are displayed. Symbols corresponding to immediate data and displacement are not displayed. If the symbol name exceeds 50 characters, the nest is displayed without the front parts. When abbreviated forms are defined, the abbreviated symbol names appear in the display. When multiple symbols are defined at the same address, only one of them will appear in the display.

1.12 Assembly Function

1.12.1 Overview

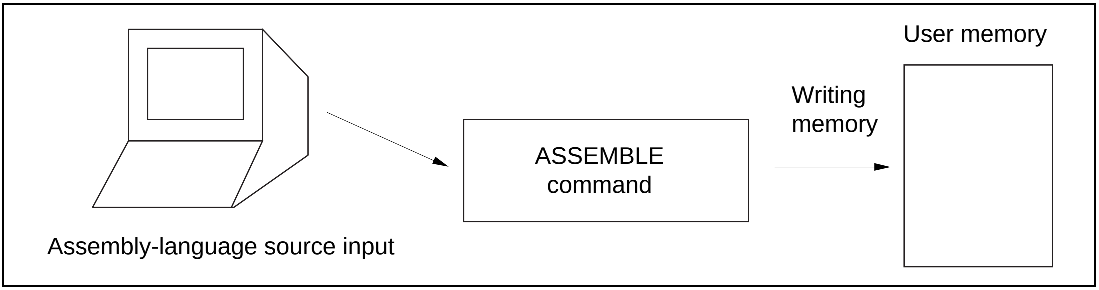


Figure 1.22 Assembly Function

The ASSEMBLE command enables line assembly as shown in figure 1.22.

- Line assembly: Assembly-language source code is input from the console and assembled one line at a time.

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

1.12.2 Input Format

The basic instruction format is as follows.

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

<label name>: A label name is a character string of up to 32 characters. Any character used in symbol names can be used, but lower case characters cannot be used. The label name must start in the first column.

<instruction mnemonic>: Any instruction mnemonic described in the SH7000-Series Programming Manual and any assembler directive listed in table 1.8 can be used.

<operand>: Any mnemonic described in the SH7000-Series Programming Manual can be used (table 1.9).

<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. If no instruction operation size is specified, byte (B) is assumed.

2. Continuation lines cannot be input.

3. The default for radix of constants is set by the RADIX command. Note that, if the specified radix is hexadecimal (specified by H), the characters A to F are considered to be numbers and cannot be used in a label name without other alphabetic characters.

Table 1.8 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 <value> times the unit length given by s: B (byte), W (word) or L (long word). 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 an area whose size is equal to <value> times the unit length given by s: B (byte), W (word) or L (long word). Default size is L.. • Up to 32,767-byte area can be reserved at one time.

Table 1.9 Operand Description

Format	Addressing Mode	Remarks	
Rn	Register direct	Rn:	General-purpose 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:	Register of the sum of products
MACL		MACL:	Register of the sum of products
PR		PR:	Procedure register
@Rn	Register indirect	Rn:	General-purpose register name
@Rn+	Register indirect with post-incrementation	Rn:	General-purpose register name
@-Rn	Register indirect with pre-decrementation	Rn:	General-purpose register name
@(disp, Rn)	Register indirect with displacement	disp:	Displacement value
		Rn:	General-purpose register name
@(R0, Rn)	Register indirect with index	R0,Rn:	General-purpose register name
@(disp, GBR)	GBR indirect with displacement	disp:	Displacement value
		GBR:	Global base register
@(R0, GBR)	GBR indirect with index	R0:	General-purpose 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, BSR instructions)
#imm	Immediate	imm:	Immediate value

Notes: 1. For the address value, immediate value and displacement values, the symbols or 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.

1.12.3 Definition of Label Names as Symbols

The names of labels in the assembly-language code are all registered as symbols, which may be used in each command. In line assembly, a symbol name cannot be defined if this has already been done.

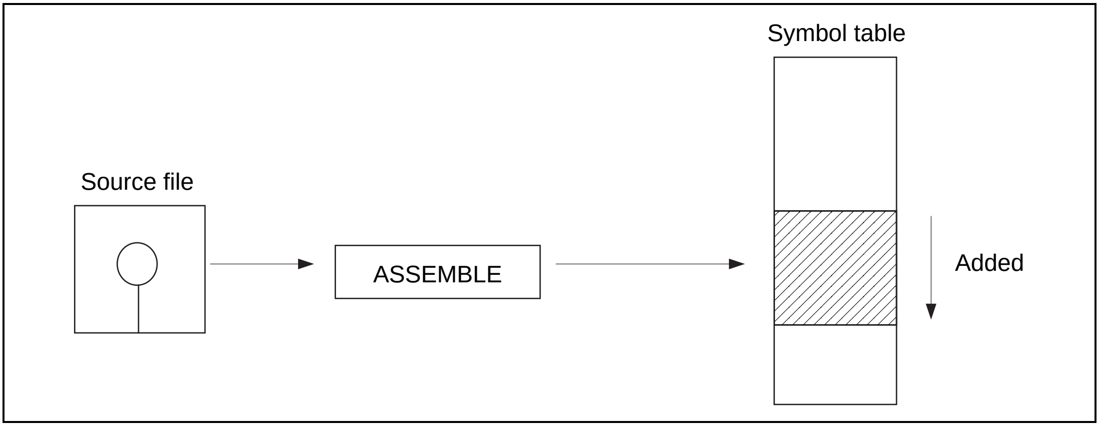


Figure 1.23 Label Name Definition

The maximum number of labels is 1,024. However, if symbols have been already registered, this figure is smaller.

1.12.4 Label Name Reference

A label name can be referenced by specifying it in an operand field in an assembly-language source statement. Label names are assembled in the following addressing modes:

- PC relative addressing mode: For instructions BRA, BT, BF, and BSR
- Immediate data
- Displacement

Forward reference is possible, but if 128 or more forward-reference labels remain unresolved, an error occurs. Label names can be referenced without ! within one ASSEMBLE command execution, but labels previously defined with the ASSEMBLE command must be preceded by ! when referenced in another ASSEMBLE command. Moreover, ! must always be used for 3-bit immediate data. In this case, symbol value must be already registered. When a disassembled program is displayed, immediate data and displacement are not displayed in symbol form.

1.12.5 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 7.2.14, DISASSEMBLE.

Section 2 Differences between the SH-2 and the Emulator

When the emulator system is initiated, the emulator resets the SH-2 as a result of a command such as switching the clock, or the RESET command is used, note that the general-purpose registers and part of the control registers are initialized.

Table 2.1 Differences between SH-2 and Emulator

Status	Register	Emulator	SH-2
Emulator initiation (power-on)	PC	Power-on reset vector value	Power-on reset vector value
	R0 to R14	H'00000000	Undefined
	R15 (SP)	Power-on reset vector value	Power-on reset vector value
	SR	H'000000F0	Undefined
	PR	H'00000000	Undefined
	VBR	H'00000000	Undefined
	GBR	H'00000000	Undefined
	MACH	H'00000000	Undefined
	MACL	H'00000000	Undefined
Reset with command	PC	Power-on reset vector value	Power-on reset vector value
	R15 (SP)	Power-on reset vector value	Power-on reset vector value
	SR	The value before reset (I bits = B'1111)	Power-on reset vector 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. Therefore, the user system hardware should be adjusted accordingly. Refer to section 4, User System Interface.

Data in cache hit cycles can be traced using this emulator. However, by doing so, chip access speed will be slower than the real SH-2 chip(s) because the external bus must be used.

- Emulation with One SH-2 Chip in Operation

  Chip access speed will be about 1 to 1.5 times slower than the real SH-2 chips depending on the cache hit rate and how often the external bus is accessed. When chip access speed is critical for emulation, disable trace acquisition in the cache hit cycles by specifying D in the CT option of the EXECUTION_MODE command as shown below. Note that trace in cache hit cycles is enabled at shipment.

(Example)	:EXECUTION-MODE_CT=E (RET)	Trace acquisition is enabled in cache hit cycles
	:EXECUTION-MODE_CT=D (RET)	Trace acquisition is disabled in cache hit cycles

For more details, refer to section 7.2.17, EXECUTION_MODE in Part III, Emulator Function Guide.

- Emulation with Two SH-2 Chips in Operation

  Chip access speed will be slower than the SH-2 chip(s) depending on the cache hit rate and how often the SH-2 bus is accessed to and from each SH-2 chip. When access speed is critical for emulation, disable the trace acquisition in the cache hit cycles by specifying D in the CT option of the EXECUTION_MODE command.

The user break controller(s) of the SH-2 chip(s) cannot be used with this emulator.

Section 3 SH-2 Function Support

The SH-2 has seven operating modes. The emulator can operate in all the MCU operating modes. However, no crystal oscillator can be connected to the emulator.

3.1 Operating Mode Setting

The user selects the clock operating mode, CS0 area bus width and master/slave mode, for the emulator with the MODE command. Table 3.1 shows the MCU operating modes, which depend on the mode setting pins (MD0 to MD2). Since no crystal oscillator can be connected to the emulator, the clock signal must be input from the user system to the EXTAL or CKIO pin. The user can access the mode setting pin status, but this does not affect the clock operating mode. When using one of clock modes 4 to 6, supply the clock signal from the user system. In these modes, the emulator cannot operate with the emulator internal clock.

Table 3.1 MCU Operating Modes

Operating Mode	MD2	MD1	MD0	SH7604
Mode 0	0	0	0	Inputs an external clock from the EXTAL pin and uses it to generate an internal clock with the same phase in the PLL1 circuit. The clock input to the PLL1 circuit will be output from the CKIO pin.
Mode 1	0	0	1	Inputs an external clock from the EXTAL pin and uses it to generate an internal clock with a phase shift of 90 degrees in the PLL1 circuit. The clock input to the PLL1 circuit will be output from the CKIO pin.
Mode 2	0	1	0	Inputs an external clock either from a crystal oscillator connected to the EXTAL and XTAL pins or from an external signal applied to the EXTAL pin, and multiplies the frequency by 1, 2, or 4 in the PLL2 circuit. A clock with the same phase as the internal LSI clock will be output from the CKIO pin.
Mode 3	0	1	1	Inputs an external clock either from a crystal oscillator connected to the EXTAL and XTAL pins or from an external signal applied to the EXTAL pin, and multiplies the frequency by 1, 2, or 4 in the PLL2 circuit. The CKIO pin will be in the high-impedance state.
Mode 4	1	0	0	Inputs a clock from the CKIO pin with the same frequency as the desired operating frequency and generates an internal clock with the same phase in the PLL1 circuit.
Mode 5	1	0	1	Inputs a clock from the CKIO pin with the same frequency as the desired operating frequency and uses it to generate an internal clock with a phase shift of 90 degrees in the PLL1 circuit.
Mode 6	1	1	0	Operates at the same frequency as the clock input from the CKIO pin. The PLL1 circuit will not operate.

Table 3.2 CS0 Area Bus Width Selection

MD4	MD3	CS0 Area Bus Width
0	0	8 bits
0	1	16 bits
1	0	32 bits

Table 3.3 Master Mode/Slave Mode Selection

MD5	Master/Slave Mode
0	Master mode
1	Slave mode

In the emulator, the operating mode previously set is saved to the configuration file of the emulator system disk (configuration file on the directory where the E7000PC system program is installed in the E7000PC). When initializing with this disk, the emulator initiates the system with the operating mode specified with the MODE command. When the operating mode is set with the MODE command, the emulator system program terminates and must be restarted.

3.1.1 Note on Using SH-2 Emulator in Multiple Emulator Configuration

When using two emulators at the same time, note the following.

System Configuration: Figure 3.1 shows the system configuration when multiple emulators are used. Each emulator pod must be specified in master or slave mode.

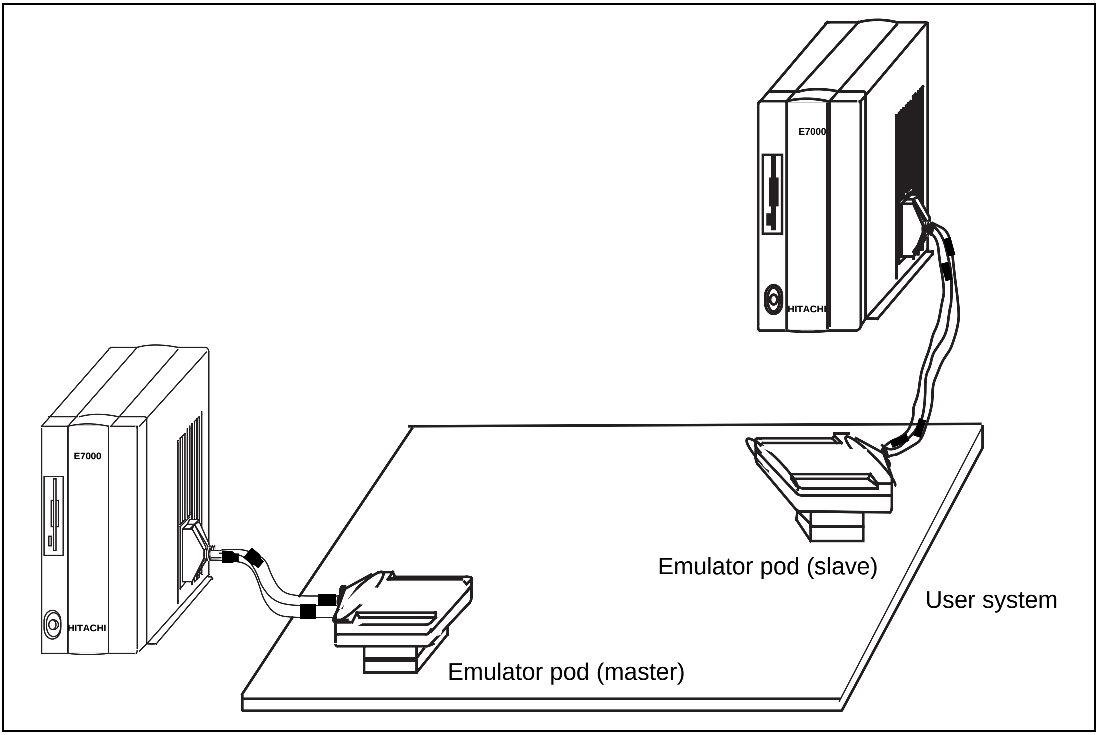


Figure 3.1 Multiple Emulator Configuration

Note on Emulator Initiation: When slave mode is specified by setting the mode selection pin MD5 to 1, the following message is displayed and the emulator cannot be initiated in master mode. To initiate the emulator in master mode, clear the mode selection pin MD5 to 0.

```
SH7604 E7000 (HSxxxxEPDxxSF) Vn.m
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
LAN IP ADDRESS FILE LOADING
MODE CHECK
USER MODE = SLAVE
:
```

Operating Mode Settings: The emulator operating mode at initiation can be specified by one of the following methods.

- Specifying the MD5 to MD0 bits in the configuration file
- Specifying the mode selection pins MD5 to MD0 on the user system
- Specifying master or slave mode with the mode selection pin MD5 and specifying other modes by the configuration file.

At emulator shipment, emulator operating mode at initiation is specified by the MD5 to MD0 bits in the configuration file. To initiate the emulator in the operating mode specified with the mode selection pins on the user system, specify the MODESET option appropriately in the MODE command as follows:

```
:MODE ; C(RET)
 E7000 MODE(MD5-0)=XX  ?    2E (RET)
 MODE SET   (C: CONFIGURATION/U: USER /M:MASTER-SLAVE) = x  ? (a) (RET)
 CONFIGURATION WRITE OK (Y/N) ? Y (RET)
```

(a) C: Operating mode is set to the mode specified in the configuration file
U: Operating mode is set to the mode specified with operating mode selection pins MD0 to MD5 on the user system
M: Master or slave mode is selected by mode selection pin MD5 and the other settings are specified with the configuration file

Notes: 1. If MODE SET = C, the emulator cannot be initiated in master mode while the mode selection pin on the user system is set to slave mode. However, if the mode selection pins MD5 to MD0 are all high (H'3F), the emulator operating mode at initiation is set by the configuration file. In this case, care must be taken to set two emulators in different modes.

2. When master or slave mode is specified by mode selection pin MD5, care must be taken to set the two emulators in different modes.

- Example

  The following shows an example of both the master and slave emulators' configuration files. In this case, specify master or slave mode with the MODE command as appropriate before connecting the emulator pod to the user system.

  — Operating mode setting in the master emulator pod

```
:MODE ; C(RET)
 E7000 MODE(MD5-0)=XX  ?    0E (RET)
 MODE SET   (C: CONFIGURATION/U: USER /M:MASTER-SLAVE) = x  ? C  (RET)
 CONFIGURATION WRITE OK(Y/N) ? Y (RET)
```

  — Operating mode setting in the slave emulator pod

```
:MODE ; C(RET)
 E7000 MODE(MD5-0)=XX  ?    2E (RET)
 MODE SET   (C: CONFIGURATION/U: USER /M:MASTER-SLAVE) = x  ? C  (RET)
 CONFIGURATION WRITE OK(Y/N) ? Y (RET)
```

  After the configuration files for master and slave emulators have been prepared, connect the emulator pods to the user system as shown in figure 3.1.

Bus Grant Signal Output Control: To operate the slave emulator pod, the bus grant signal (BGR) must be provided from master emulator pod to the slave emulator pod. Note, however, that when the master emulator pod outputs the bus grant signal, the master emulator pod is a little slower when operating in master mode than when operating alone. Thus, the bus grant signal output must be controlled by the BRQ option of the EXECUTION_MODE command as described below, according to the priority of master or slave emulator pod to be debugged.

- BRQ option setting

  E: Enables the bus grant signal output both in E7000 operation and program execution.

  M: Enables the bus grant signal output in program execution but disables bus grant signal output during E7000 operation.

  D: Disables the bus grant signal output both in E7000 operation and program execution.

Cache Hit Cycle Trace: To acquire trace information for cache hit cycles, the emulator uses the bus. This can degrade another emulator's operation speed when using emulators in multiple emulator configuration. The CT option of the EXECUTION_MODE command can be used to disable trace information acquisition during cache hit cycles.

- CT option setting
  E: Enables trace information acquisition during cache hit cycles
  D: Disables trace information acquisition during cache hit cycles

Multibreak: By using the external probe pin, the user can cause a break in both master and slave emulator pods simultaneously. For details, refer to section 1.4.1, Hardware Break.

Other Precautions: If the SLEEP instruction is executed in the emulator, timeout (TOUT) is displayed. In a multiple emulator configuration, however, either BREQ input from an emulator or a timeout (TOUT) is displayed. In addition, the emulation memory in one emulator is independent of that in another emulator. Accordingly, the emulation memory in one emulator cannot be accessed from another emulator.

3.2 Memory Area

The SH-2 has a maximum memory area of 128 Mbytes × 4 areas, to which areas standard emulation memory (512 kbytes) can be set in units of 128 kbytes. Areas that are not set as emulation memory are set as user system memory.

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

The user can specify write-protected and access-prohibited areas as emulation memory. The access cycle for emulation memory is as follows.

- Standard emulation memory: No wait

Write-protected areas and access-prohibited (guarded memory) areas can be allocated to user memory in units of 128 kbytes. Note that a write-protected area and a guarded memory area can be allocated to the external memory area in the user memory area.

- W: Write-protected
- G: Access-prohibited (guarded)

The above attribute settings are valid in external memory only. Refer to section 7.2.23, MAP, for details on attribute settings.

3.2.1 Internal I/O Area

When the internal I/O area is accessed, the emulator accesses the SH-2 internal I/O, regardless of the MAP command setting. The user can read and write to the internal I/O area from the user program or with emulator commands. When writing to the internal I/O area with an emulator command (MEMORY), the following warning message is displayed and the emulator starts writing without verifying.

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

However, the user cannot write to the internal I/O with the FILL command.

3.2.2 External Memory Area

The SH-2 external memory area can be allocated to all memory attributes that the emulator supports. Memory corresponding to the allocated attributes can be accessed by either the user program or emulator commands.

3.3 Low-Power Consumption Mode (Sleep and Standby)

For reduced power consumption, the SH-2 has sleep and standby modes.

The sleep and standby modes are switched using the SLEEP instruction. These modes can be cleared with either the normal clearing function or with the break condition satisfaction (including (BREAK) or (CTRL) + C key input), and the program breaks. Trace information is not acquired in sleep and standby modes.

Notes: 1. When restarting after a break, the user program will restart at the instruction following the SLEEP instruction.

2. During sleep mode, if the user accesses or modifies the memory in parallel mode, the sleep mode is cleared and the user program execution continues from the instruction following the SLEEP instruction.

3.4 Interrupts

During emulation, the user can interrupt the SH-2. If an interrupt occurs while the emulator is waiting for command input, the interrupt is not processed. However, if an edge sensitive 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.

3.5 Control Input Signals (RES, WAIT, BREQ)

The SH-2 control input signals are RES, WAIT and BREQ. The RES signal is valid only when emulation has been started with the GO command. The WAIT and BREQ signals are valid during execution with either the GO command or the STEP command. Therefore, while the emulator is waiting for command input, the user cannot input RES, WAIT or BREQ signals to the SH-2. BREQ signals can be masked using the EXECUTION_MODE command.

Note that the BREQ signal means signals input to the BACK/BRLS pin, not signals output from the BREQ/BGR pin, in this emulator.

3.6 Watchdog Timer (WDT)

The WDT only operates during emulation (GO, STEP), and does not operate when the emulator is waiting for command input. The timer stops at a break and restarts when emulation recommences.

3.7 16-Bit Free-Running Timer (16-Bit FRT)

The 16-bit FRT operates during the command input wait state as well as during emulation. Even after the user program has stopped, when a break condition is satisfied after the user program has been started with a GO command, the 16-bit FRT continues to operate. Therefore, the timer pins are valid even when the user program has stopped. The user can rewrite the timer registers with the MEMORY command.

3.8 Serial Communications Interface

The serial communications interface signals are connected to the user system directly from the SH-2 on the pod. Therefore, the interface is valid during the command input wait state as well as emulation. For example, when writing data to the transmit data register (TDR) with the MEMORY command, after the serial communications interface output has been prepared, data is output to the TXD line.

3.9 Direct Memory Access Controller (DMAC)

The DMAC operates during emulation execution and the command input wait state. When a transfer request occurs, the emulator carries out DMA transfer.

3.10 Bus State Controller

The SH-2 wait state controller has a programmable wait mode and a WAIT pin input mode. While the programmable wait mode is valid when the emulation memory or user external memory is accessed, input to the user WAIT pin is valid only 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. The refresh cycle controller operates continuously when the emulator is carrying out SDRAM/DRAM/PSRAM refresh control, even during the command input wait state.

Section 4 User System Interface

The emulator is connected to the user system directly. Probe signal trace and break can be enabled by connecting the external probe to the user system.

The trigger output probe can output a low-level pulse as an oscilloscope trigger signal. For details, refer to section 1.7, Trigger Output.

User System Interface Circuits: The circuits that interface the SH-2 in the emulator 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. Table 4.1 shows the bus timing when using the emulator. Figures 4.1 and 4.2 show the basic bus cycle timing and the user interface circuits, respectively.

Table 4.1 Bus Timing (at 28.5-MHz Operation)

Parameter	MCU Specification (ns)	Emulator Specification (ns)
tAD1	3 (min)	16.4 (typ)
tAD2	14 (min)	28.8 (typ)
tBSD1	3 (min)	12.0 (typ)
tBSD2	14 (min)	21.6 (typ)
tCSD1	3 (min)	13.2 (typ)
tRWD1	3 (min)	10.8 (typ)
tWED	3 (min)	12.4 (typ)
tWDD1	3 (min)	18.0 (typ)
tWDD2	14 (min)	28.8 (typ)
tWDH1	3 (min)	16.0 (typ)
tWDH2	14 (min)	57.6 (typ)
tRASD1	14 (min)	20.0 typ)
tRASD2	3 (min)	12.4 (typ)
tCASD1	14 (min)	18.8 (typ)
tCASD2	3 (min)	12.0 (typ)
tDQMD	14 (min)	22.0 (typ)

Note: Adjust user system hardware considering the above bus timimg conditions.

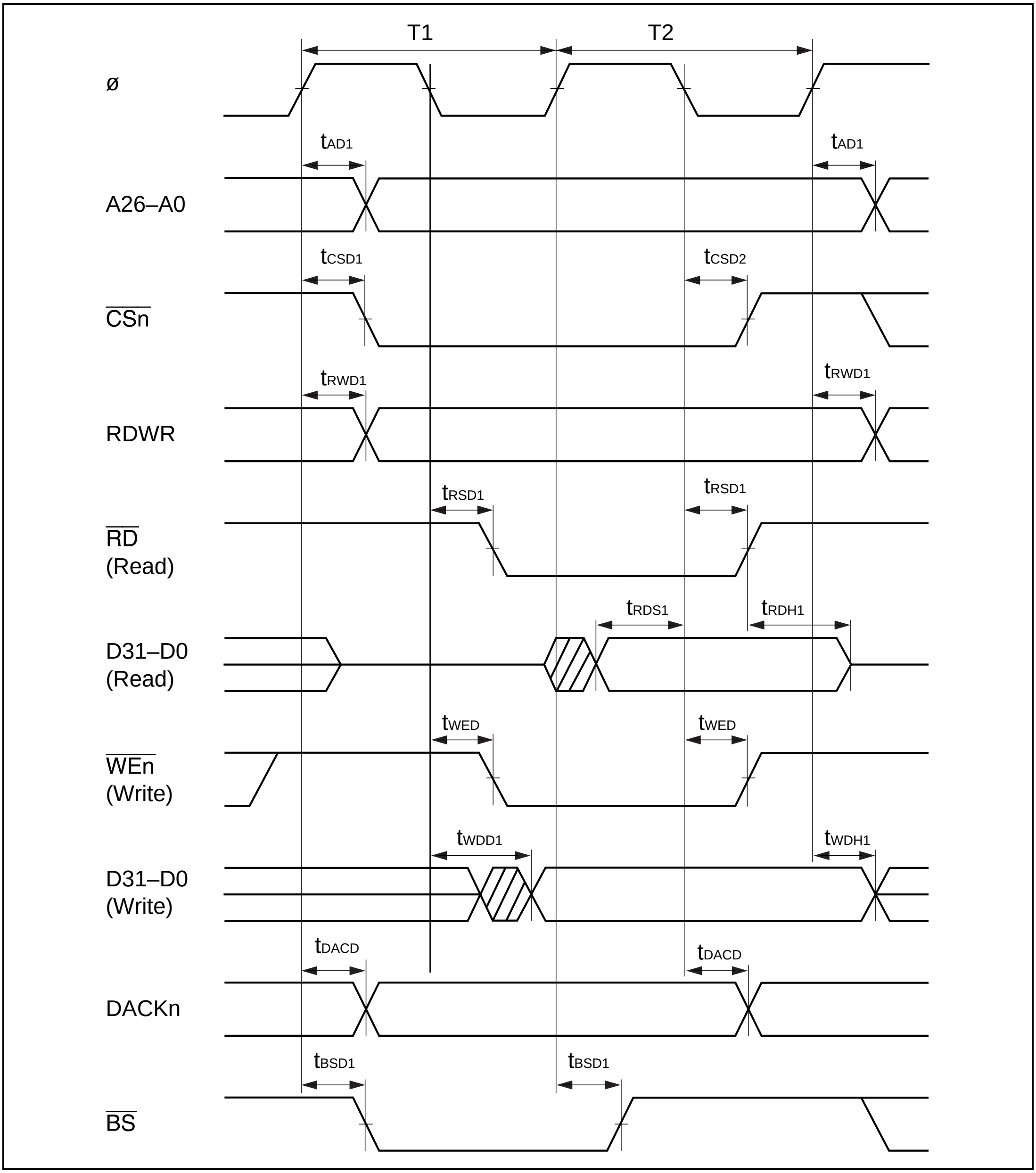


Figure 4.1 Basic Bus Cycle Timing (Two States)

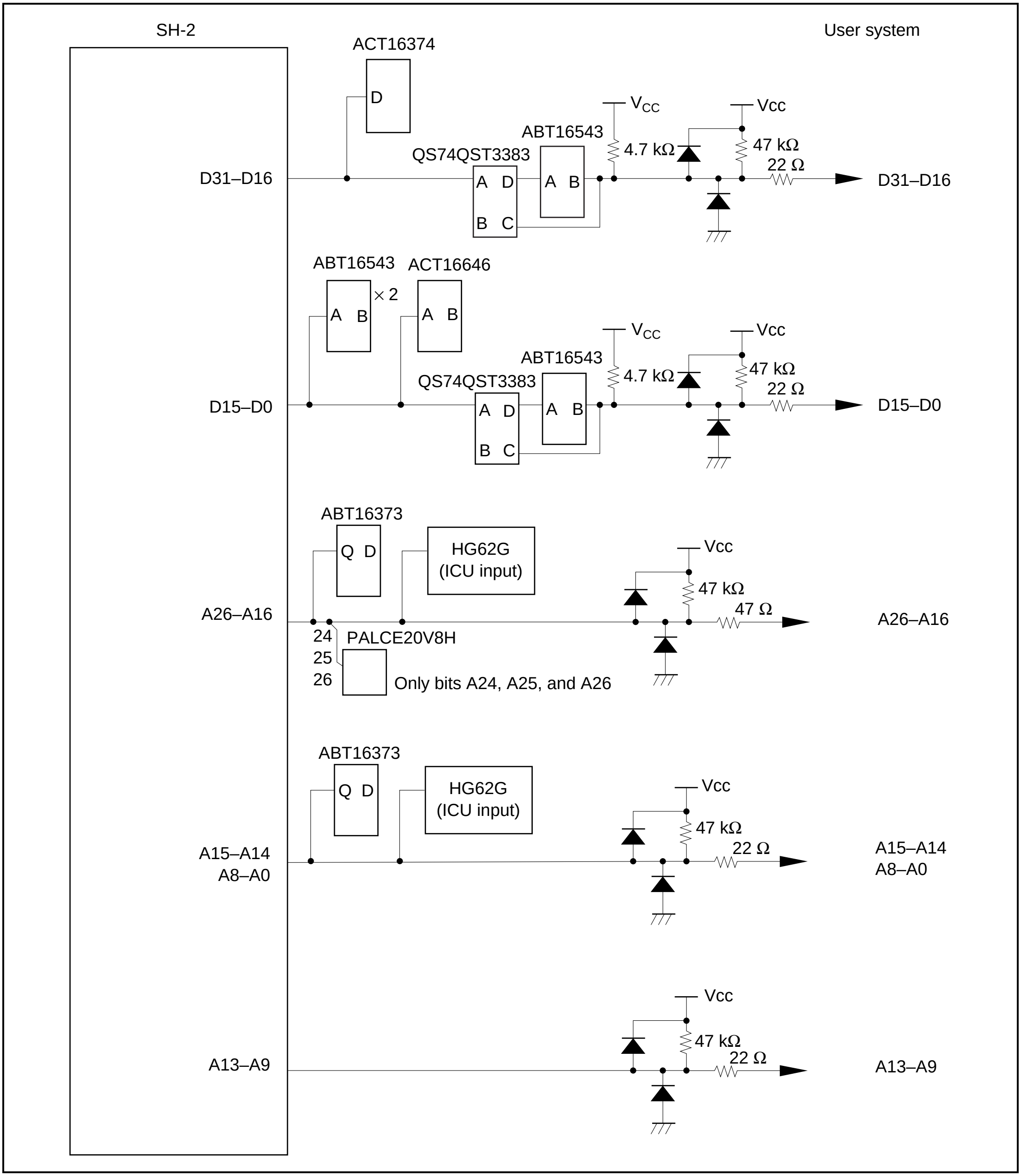


Figure 4.2 User System Interface Circuits

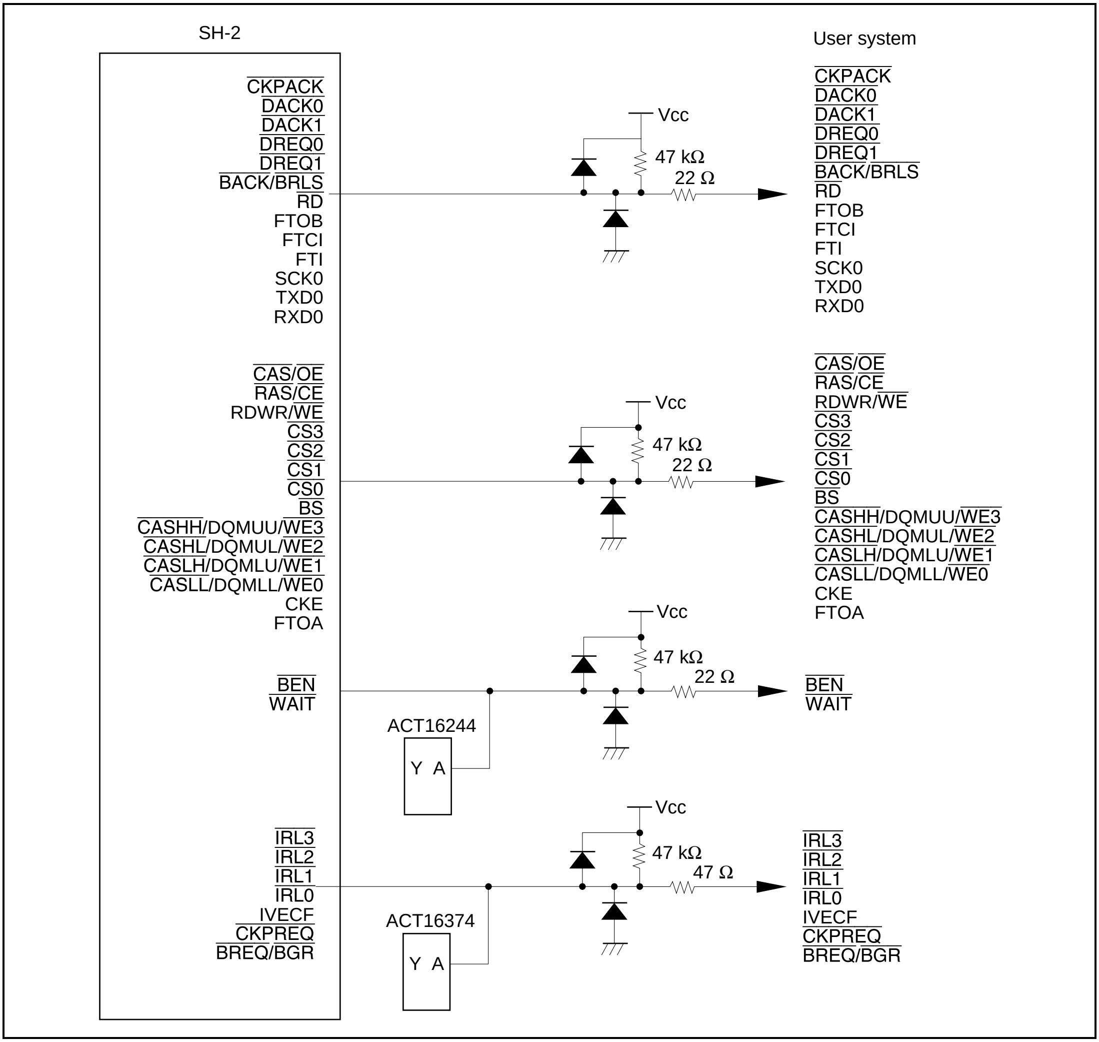


Figure 4.2 User System Interface Circuits (cont)

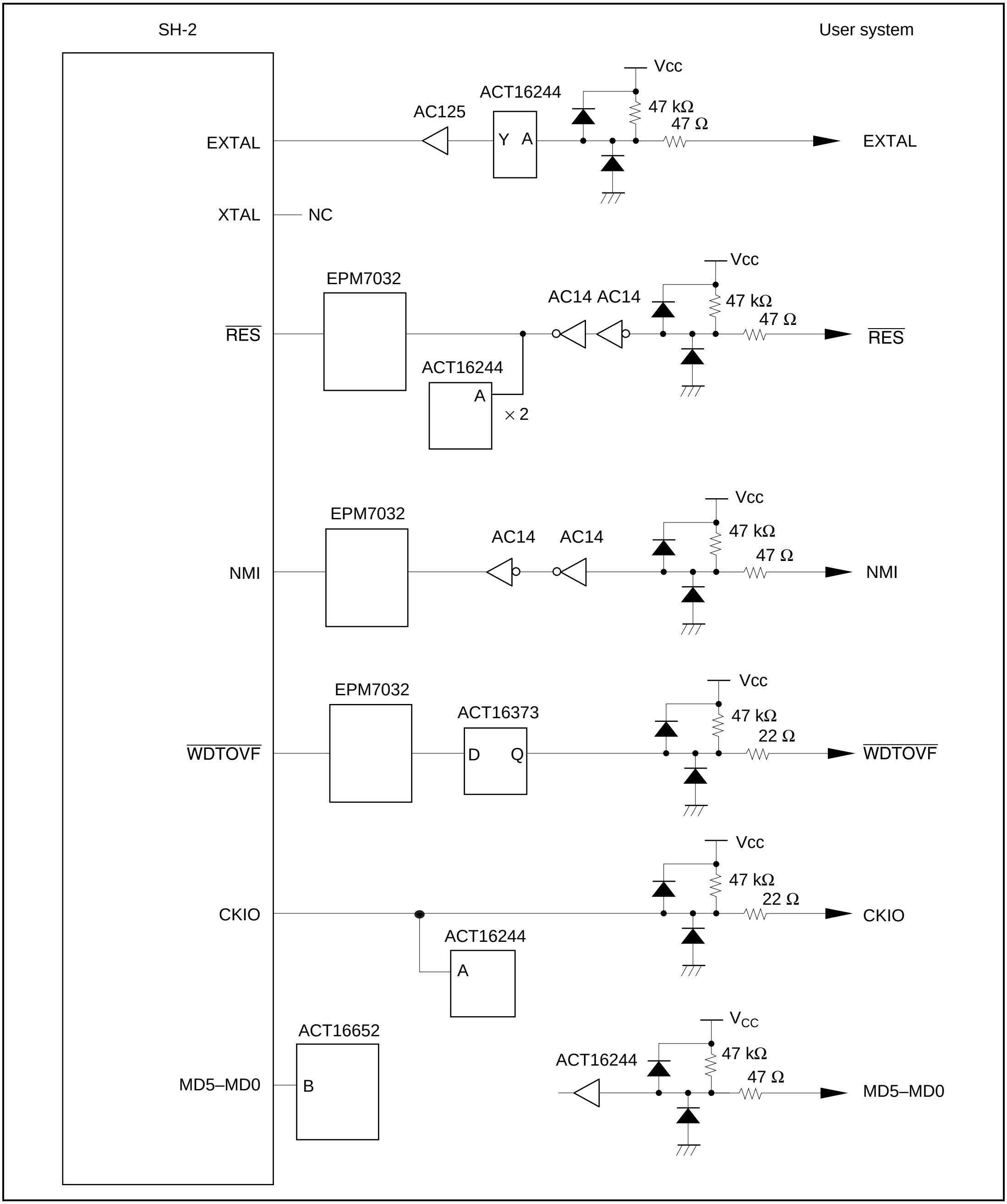


Figure 4.2 User System Interface Circuits (cont)

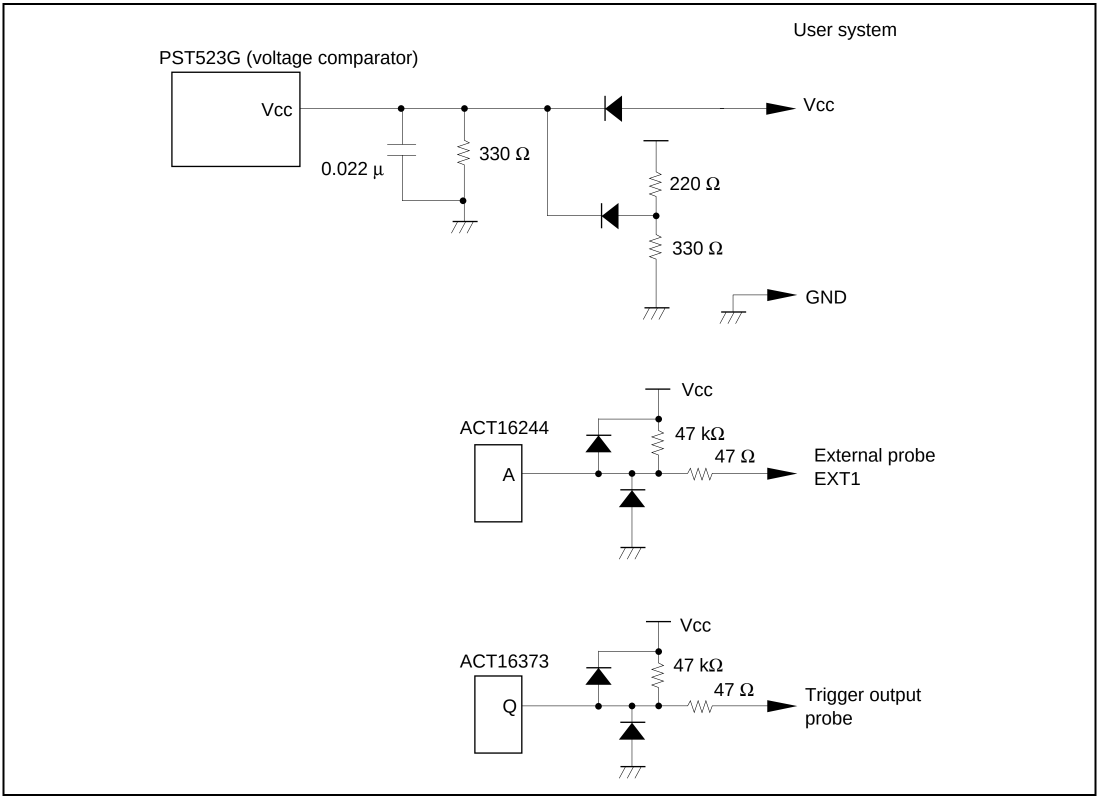


Figure 4.2   User System Interface Circuits (cont)

Section 5 Troubleshooting

The emulator internal system test checks the emulator's internal RAM and registers at power-on (emulator monitor initiation) and at system program initiation.

Section 5.1 describes the emulator internal system test when using the E7000 emulator (HS7000EST01H), and section 5.2 describes the test when using the E7000PC emulator (HS7000ESTP1H).

5.1 Internal System Test Using the E7000

Internal System Test at Power-On: The E7000 checks its internal RAM and registers at power-on. While tests are in progress, the following messages are displayed:

```
E7000 MONITOR  Vn.m
Copyright (C) 1991 Hitachi, Ltd.                (a)
Licensed Material of Hitachi, Ltd.

TESTING                                         (b)
RAM 0123

**  E7000 SYSTEM LOADING **                     (c)
*** FD NOT READY ***

START E7000
S : START E7000
R : RELOAD & START E7000
B : BACKUP FD                                   (d)
F : FORMAT FD
L : SET LAN PARAMETER
T : START DIAGNOSTIC TEST
     (S/R/B/F/L/T) ?
```

(a) E7000 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=xxxxxxxxc
  ```

- After RAM testing is completed, the registers are tested. If an error occurs, the following message is displayed:

  ```
  *** xxxx REGISTER ERROR W-DATA=xx  R-DATA=xx
  ```

  xxxx: Name of E7000 internal register where an error occurs

(c) E7000 system program is being loaded. If the E7000 system disk is not inserted, the E7000 monitor enters command input wait state and the following message is displayed.

```
***  FD NOT READY
```

(d) The E7000 monitor is in command input wait state.

Note: Operation continues if an error occurs in step (b) or (c), but the error should be investigated according to section 5.2, Troubleshooting Procedure, without loading the E7000 system program.

Internal System Test at E7000 System Program Initiation: The E7000 system performs internal system tests, mainly on the E7000 registers, at its initiation.

```
**   E7000 SYSTEM LOADING   **

SH7604 E7000   (HS7604EPD70SF)   Vn.m
Copyright (C)   Hitachi, LTD. 1993                (a)
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING                        (b)
LAN IP ADDRESS FILE LOADING                       (c)
MODE CHECK                                        (d)
HARD WARE REGISTER  READ/WRITE CHECK              (e)
POD SYSTEM LOADING                                (f)
EMULATOR POD TEST                                 (g)
**  RESET IN BY E7000 !                           (h)
CLOCK = USER

MASTER MODE = XX   (MD5-0=XX)                     (i)
MODE SET = X

FAILED AT xxxx                                    (j)
REMAINS EMULATION MEMORY S=D'0512kB               (k)

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

(a) E7000 system program start message. Vn.m indicates the version and revision numbers.

(b) Configuration file is being loaded. If an invalid configuration file is assigned, the following message is displayed:

*** 9:INVALID CONFIGURATION FILE

If no configuration file is contained in the system disk, the following message is displayed:

*** 10:CONFIGURATION FILE NOT FOUND

At this time, an unavailable system disk is inserted. Change system disks and re-execute.

(c) IP address information file for the host system connected via the LAN interface is being loaded. If an invalid IP address information file is assigned, the following message is displayed.

```
LAN  I/O  ERROR  (E0xx)
socket  library  error  n: <error message>
```

xx: Current processing (refer to table 12.6 in section 12, Error Messages)
n: Error code (refer to table 12.5 in section 12, Error Messages)
<error message>: Refer to table 12.5 in sectio

n 12, Error Messages

If an error message that is not shown in table 12.6 in section 12, Error Messages is displayed, the error may have occurred in the host system connected via the FTP interface. Also check the host system.

(d) The specified E7000 mode is compared with the mode selection pin status on the user system. If the E7000 is in master mode and the mode selection pin status is in slave mode, the SH-2 operating mode is not set to the mode specified in the configuration file. In this case, initiation is terminated and the following message is displayed:

```
USER MODE = SLAVE
```

Change the E7000 mode to slave mode with the MODE command or change the mode selection pin status to master mode.

(e) The E7000 control registers are being checked. If an error occurs, one of the following messages is displayed.

```
**   xxx   REGISTER ERROR W-DATA = xxxx R-DATA = xxxx                                (i)
**   BREAK MEMORY ERROR ADDR = xxxx  W-DATA = xxxx   R-DATA = xxxx                   (ii)
**   SHARED RAM ERROR ADDR = xxxxxx  W-DATA = xxxxxxxx   R-DATA = xxxxxxxx           (iii)
```

(i) A write verification error occurred in one of the following E7000 internal registers: RAR, TCR, BQR, TSR, TBM
(ii) An error occurred in break memory
(iii) An error occurred in the shared RAM

(f), (g) The emulator pod is being tested. If an error occurs, the following message is displayed.

```
*** INVALID  EMULATOR  POD!                        (i)
*** EMULATOR  POD  NOT  READY !                    (ii)
*** EMULATOR  POD  ERROR  CODE=xx                  (iii)
*** EMULATOR  POD  SYSTEM FILE NOT FOUND           (iv)
```

(i) The wrong MCU emulator pod is connected. Please check the MCU type and use the appropriate E7000 system program, or exchange the emulator pod.

(ii) The emulator pod is not connected correctly. Connect the emulator pod to the E7000 correctly.

(iii) One of the following errors was generated:

xx: error code

RE:	An error occurred in the emulator pod work RAM
RA:	An error occurred in shared RAM
RM:	An error occurred in the MCU installed in the emulator pod
IO:	An error occurred in the internal ROM substitute RAM
IA:	An error occurred in the internal RAM substitute RAM
BK:	The BREAK key was pressed, suspending operation

(iv) System programs for the emulator pod are not loaded because the wrong E7000 system program is installed. Insert the correct E7000 system disk and restart the E7000.

Note: If the (CTRL) + C keys or (BREAK) key is pressed during emulator pod testing, the test is suspended (error code BK).

(h) The MCU is reset, the clock is set, and the specified clock type is displayed ((h) is not executed if an error has occurred in step (d), (e), (f), or (g)).

(i) The MCU operating mode on the E7000, the status of user system mode selection pins, and the size of internal ROM and RAM are displayed.

(j) MCU pins are being checked. For details, refer to section 7.2.8, CHECK ((j) is not executed if an error has occurred in step (d), (e), (f), or (g)).

(k) The remaining emulation memory size that can be assigned.

(l) The emulator pod system program is initiated.

E7000 System Failure: If an invalid exception occurs during E7000 monitor or E7000 system program execution, the system shuts down, and the following message is displayed:

```
<exception>    PC=xxxxxxxx
***  E7000  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 Internal System Test Using the E7000PC

Internal System Test at Power-On: The E7000PC checks its internal RAM and registers at power-on. While tests are in progress, the following messages are displayed:

```
E7000 MONITOR  Vn.m
Copyright (C) 1991 Hitachi, Ltd.                  (a)
Licensed Material of Hitachi, Ltd.

TESTING                                           (b)
RAM 0123

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

(a) E7000PC 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=xxxxxxxxc

- After RAM testing is completed, the registers are tested. If an error occurs, the following message is displayed:

  *** xxxx REGISTER ERROR W-DATA=xx R-DATA=xx

  xxxx: Name of E7000PC internal register where an error occurs

(c) The E7000PC monitor is in command input wait state.

Note: Operation continues if an error occurs in step (b) or (c), but the error should be investigated according to section 5.2, Troubleshooting Procedure, without loading the E7000PC system program.

Internal System Test at E7000PC System Program Initiation: The E7000PC system performs internal system tests, mainly on the E7000PC registers, at its initiation.

```
**   E7000 SYSTEM LOADING   **

SH7604 E7000   (HS7604EPD70SF)   Vn.m
Copyright (C)   Hitachi, LTD. 1993                (a)
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING                        (b)
LAN IP ADDRESS FILE LOADING                       (c)
MODE CHECK                                        (d)
HARD WARE REGISTER  READ/WRITE CHECK              (e)
POD SYSTEM LOADING                                (f)
EMULATOR POD TEST                                 (g)
**  RESET IN BY E7000 !                           (h)
CLOCK = USER

MASTER MODE = XX   (MD5-0=XX)                     (i)
MODE SET = X

FAILED AT xxxx                                    (j)
REMAINS EMULATION MEMORY S=D'0512kB               (k)

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

(a) E7000PC system program start message. Vn.m indicates the version and revision numbers.

(b) Configuration file is being loaded. If an invalid configuration file is assigned, the following message is displayed:

*** 9:INVALID CONFIGURATION FILE

If no configuration file is contained in the directory where the current system is stored, the following message is displayed:

*** 10:CONFIGURATION FILE NOT FOUND

This message indicates that the environment variable is not set correctly. Set the environment variable correctly and re-execute.

(c) IP address information file for the host system connected via the LAN interface is being loaded. If an invalid IP address information file is assigned, the following message is displayed.

```
LAN  I/O  ERROR  (E0xx)
socket  library  error  n: <error message>
```

xx: Current processing (refer to table 12.6 in section 12, Error Messages)
n: Error code (refer to table 12.5 in section 12, Error Messages)
<error message>: Refer to table 12.5 in section 12, Error Messages

If an error message that is not shown in table 12.6 in section 12, Error Messages is displayed, the error may have occurred in the host system connected by the FTP interface. Also check the host system.

(d) The specified E7000PC mode is compared with the mode selection pin status on the user system. If the E7000PC is in master mode and the mode selection pin status is in slave mode, the SH-2 operating mode is not set to the mode specified in the configuration file. In this case, initiation is terminated and the following message is displayed:

```
USER MODE = SLAVE
```

Change the E7000PC mode to slave mode with the MODE command or change the mode selection pin status to master mode.

(e) The E7000PC control registers are being checked. If an error occurs, one of the following messages is displayed.

```
**   xxx   REGISTER ERROR W-DATA = xxxx R-DATA = xxxx                                    (i)
**   BREAK MEMORY ERROR ADDR = xxxx  W-DATA = xxxx   R-DATA = xxxx                       (ii)
**   SHARED RAM ERROR ADDR = xxxxxx  W-DATA = xxxxxxxx   R-DATA = xxxxxxxx               (iii)
```

(i) A write verification error occurred in one of the following E7000PC internal registers: RAR, TCR, BQR, TSR, TBM
(ii) An error occurred in break memory
(iii) An error occurred in the shared RAM

(f), (g) The emulator pod is being tested. If an error occurs, the following message is displayed.

```
*** INVALID  EMULATOR  POD!                          (i)
*** EMULATOR  POD  NOT  READY !                      (ii)
*** EMULATOR  POD  ERROR  CODE=xx                    (iii)
*** EMULATOR  POD  SYSTEM FILE NOT FOUND             (iv)
```

(i) The wrong MCU emulator pod is connected. Please check the MCU type and use the appropriate E7000PC system program, or exchange the emulator pod.

(ii) The emulator pod is not connected correctly. Connect the emulator pod to the E7000PC emulator correctly.

(iii) One of the following errors was generated:

xx: error code

RE:	An error occurred in the emulator pod work RAM
RA:	An error occurred in shared RAM
RM:	An error occurred in the MCU installed in the emulator pod
IO:	An error occurred in the internal ROM substitute RAM
IA:	An error occurred in the internal RAM substitute RAM
BK:	The BREAK key was pressed, suspending operation

(iv) System programs for the emulator pod are not loaded because the wrong E7000PC system program is installed. Insert the correct E7000PC system disk and restart the E7000PC.

Note: If the (CTRL) + C keys or (BREAK) key is pressed during emulator pod testing, the test is suspended (error code BK).

(h) The MCU is reset, the clock is set, and the specified clock type is displayed ((h) is not executed if an error has occurred in step (d), (e), (f), or (g)).

(i) The MCU operating mode on the E7000PC, the status of user system mode selection pins, and the size of internal ROM and RAM are displayed.

(j) MCU pins are being checked. For details, refer to section 7.2.8, CHECK ((j) is not executed if an error has occurred in step (d), (e), (f), or (g)).

(k) The remaining emulation memory size that can be assigned.

(l) The emulator pod system program is initiated.

E7000PC System Failure: If an invalid exception occurs during E7000PC monitor or E7000PC system program execution, the system shuts down, and the following message is displayed:

```
<exception>   PC=xxxxxxxx
***  E7000  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.

5.3 Troubleshooting Procedure

This section attempts to reduce the time taken by troubleshooting by providing a troubleshooting Problem Analysis Diagram (PAD, see figure 5.1).

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 (HS7604TM70ME), and inform a Hitachi sales agency of the test results in detail because a system defect may be caused by a number of reasons.

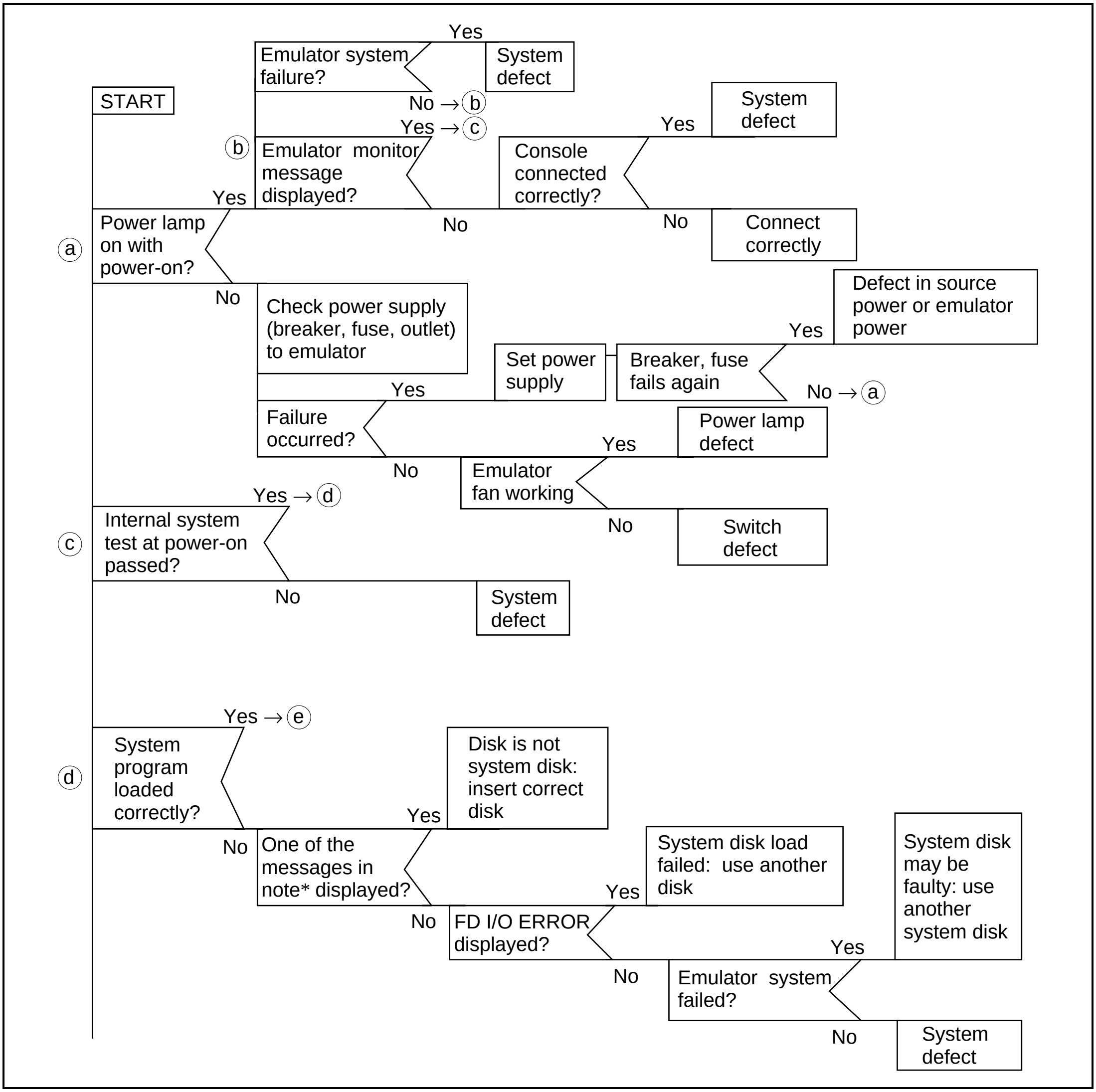
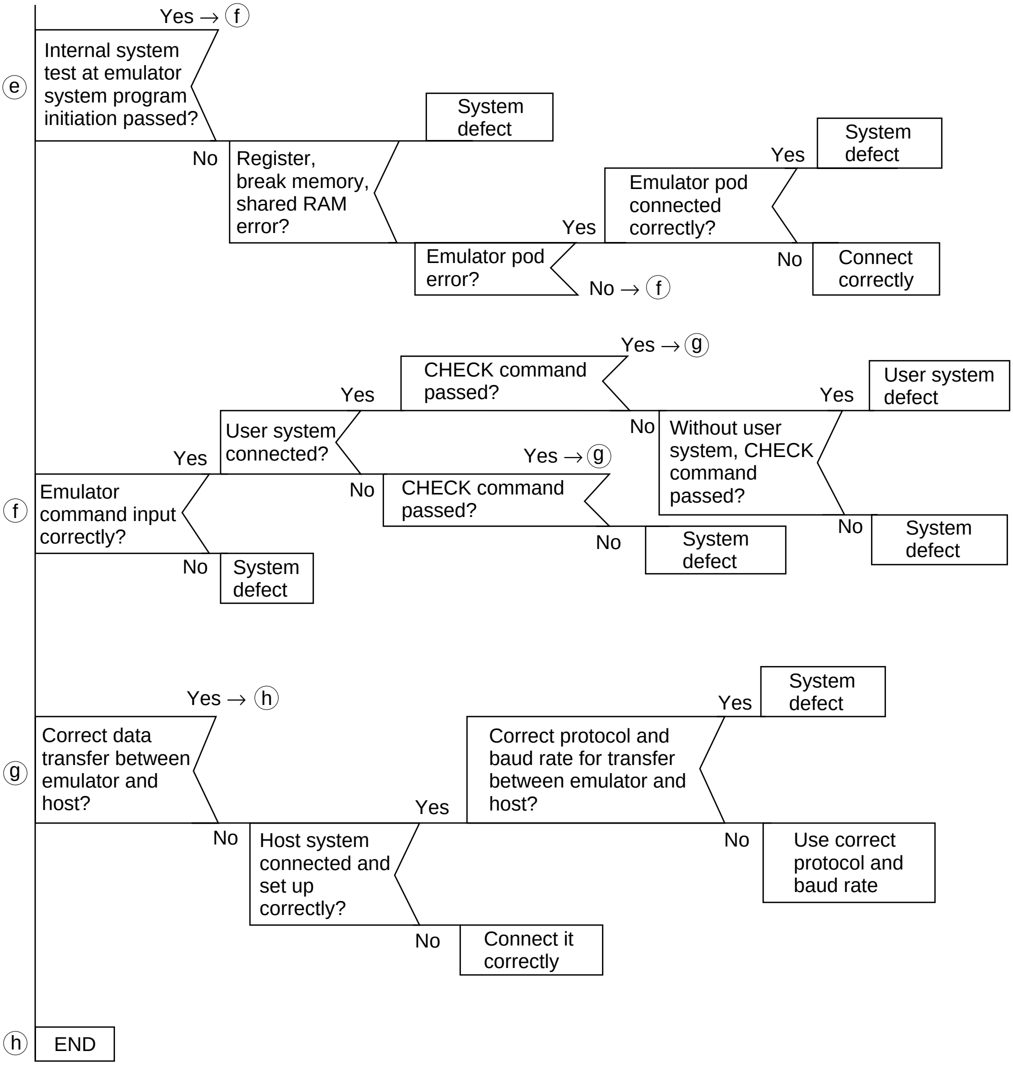

START
Power lamp on with power-on?
Yes
No
Emulator monitor message displayed?
Emulator system failure?
System defect
No → b
Yes → c
Console connected correctly?
System defect
Connect correctly
Check power supply (breaker, fuse, outlet) to emulator
Failure occurred?
Set power supply
Breaker, fuse fails again
Defect in source power or emulator power
No → a
Emulator fan working
Power lamp defect
Switch defect
Internal system test at power-on passed?
Yes → d
System defect
System program loaded correctly?
Yes → e
One of the messages in note* displayed?
Disk is not system disk: insert correct disk
FD I/O ERROR displayed?
System disk load failed: use another disk
Emulator system failed?
System disk may be faulty: use another system disk
System defect
a
b
c
d

Figure 5.1 Troubleshooting PAD


Yes → f
e
Internal system test at emulator system program initiation passed?
No
System defect
Register, break memory, shared RAM error?
Yes
Emulator pod error?
No → f
Yes
Emulator pod connected correctly?
Yes
System defect
No
Connect correctly
Yes → g
CHECK command passed?
Yes
User system connected?
Yes
f
Emulator command input correctly?
No
System defect
No
CHECK command passed?
Yes → g
No
System defect
No
Without user system, CHECK command passed?
Yes
User system defect
No
System defect
Yes → h
g
Correct data transfer between emulator and host?
No
Host system connected and set up correctly?
No
Connect it correctly
Yes
Correct protocol and baud rate for transfer between emulator and host?
Yes
System defect
No
Use correct protocol and baud rate
h
END

Note: If one of the following messages is displayed, check that the correct system disk is set:

FD NOT SYSTEM FD
CONFIGURATION FILE NOT FOUND
INVALID CONFIGURATION FILE
POD SYSTEM FILE NOT FOUND

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<parameter>;<option>  (RET)

       Δ:  Space
   (RET):  (RET) key
```

Note that each command can be specified in abbreviated form.

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 as part 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, symbols, or file names can be entered as command parameters or options. Spaces (Δ) or commas (,) can be inserted between words. Words are described below:

Constants: Numeric constants, character constants, and expression can be used as constants.

- Numeric constants

  The following shows numeric constant formats. A radix is entered at the head of a numeric constant.

  S'nnnnnnnn

  S: Radix of a constant
  B: Binary
  Q: Octal
  D: Decimal
  H: Hexadecimal (At emulator initiation)
  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. For example, ' ' ' means the character constant ' (H'27).

  Example: ' A' = H'41

  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, symbols, and operators. As an operator, + (addition) or – (subtraction) can be specified.

  Examples: D'10 + H'20 (H'2A)
  20 – 4
  –1

Symbols: There are two types of symbols; symbols defined with emulator's SYMBOL or ASSEMBLE command and that defined with an assembler or C compiler on a cross system. A basic format for each symbol type is shown below.

- Symbols defined with emulator command

  !<symbol name>

  <symbol name>: A name defined by the emulator's SYMBOL command or ASSEMBLE command. Note that only the upper case characters can be defined by the ASSEMBLE command.

- Symbols defined on a cross system

  These symbols are divided into two groups; a general symbol indicating a label name, variable name, or function name, and a line number symbol indicating an address where line number of a compiled or assembled listing file is located.

  — General symbol (starts with !)

  !<unit name>/<symbol name>/....

  — Line number symbol (starts with &)

  &<unit name>/<line number>

  Examples: !prog/main
  !prog/_sub
  &test/100

  For details, refer to section 1.11, Symbolic Debugging Function.

Notes on Using Symbols:

1. Do not use the following characters as symbols:

   ; : ( ) / + – , = ? . ! & ¥

2. A symbol name can contain up to 32 characters.

3. Only the externally defined labels used in the assembler or static symbols used in the C compiler can be defined.

4. Uppercase letters and lowercase letters are distinguished. However, in the ASSEMBLE command, a symbol in lowercase letters cannot be defined.

- File name

  A file name can be specified as a command parameter. The general file name format is as follows:

  <drive name>:<file name>.<extension>

  Note that the drive name is specified only in the FILE_COPY command when a floppy disk is copied to another floppy disk. For details, refer to section 8.2, Files.

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 abortion	CTRL + C (BREAK)	Aborts 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	Temporarily stops display. Resumes display by entering CTRL and Q keys.
• Display restart	CTRL + Q	Restarts display.
• Display the previous 16 lines	CTRL + P	Effective only for the DUMP and TRACE commands. Displays the 16 lines before the first line of the current screen and then stops. Pressing the (RET) key restarts the 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 returns 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 options again.

```
(Example)  :D 1000 1010 (RET)
           : Execution of other command
           :D.(RET)
           :D 1000 1010
```

Displays the parameters specified in the DUMP command and enters command input wait state.

6.2.4 Cursor Control and Character Editing

•	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 first position to the right of the previous space or 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 (10th + 1)th column.

Section 7 Emulation Commands

7.1 Overview

The emulator provides a wide range of functions such as break and trace. Table 7.1 lists the emulation commands that enable these functions.

Table 7.1 Emulation Commands

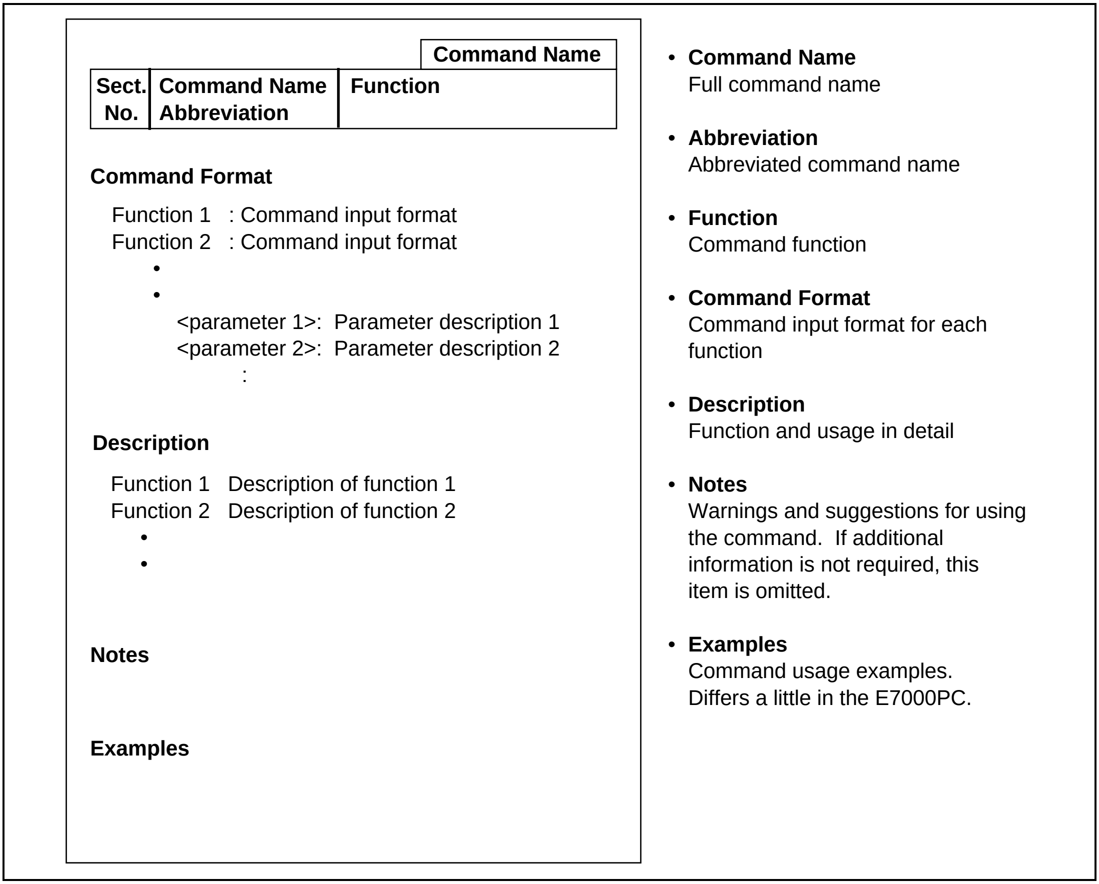
Command	Function	Usable/Unusable in Parallel Mode
.<register>	Modifies and displays register contents	Unusable
!<symbol> or &<symbol>	Displays symbol value	Usable
ABORT	Terminates emulation in parallel mode	Usable
ASSEMBLE	Assembles program	Unusable
BREAK	Sets, displays, and cancels software breakpoints	Only display function is available
BREAK_CONDITION1, 2,3,4,5	Sets, displays, and cancels hardware break condition during emulation	Unusable
BREAK_SEQUENCE	Sets, displays, and cancels software breakpoints with pass sequence specification	Only display function is available
CHECK	Tests SH-2 pin status	Unusable
CLOCK	Sets and displays clock	Only display function is available
COMMAND_CHAIN	Inputs emulator commands from a file (only for E7000)	Usable
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	Specifies and displays execution mode	Unusable
FILL	Writes data to memory	Unusable
GO	Executes realtime emulation	Unusable
HELP	Displays all commands and command format	Usable
HISTORY	Displays all commands to be input	Usable

Table 7.1 Emulation Commands (cont)

Command	Function	Usable/Unusable in Parallel Mode
ID	Displays the emulator program version number	Usable
MAP	Specifies and displays memory attribute	Unusable
MEMORY	Modifies and displays memory contents	Usable
MODE	Specifies and displays the SH-2 operating mode	Unusable
MOVE	Transfers memory contents	Unusable
MOVE_TO_RAM	Moves ROM contents to standard emulation memory	Unusable
PRINT	Sets or cancels output device for command result display (only for E7000)	Usable
QUIT	Terminates emulator system program	Unusable
RADIX	Specifies and displays radix for numeric input	Usable
REGISTER	Displays register contents	Unusable
RESET	Resets SH-2	Unusable
RESULT	Displays execution results	Unusable
SHORT_SYMBOL	Defines a short format for a symbol and displays current symbol definition	Usable
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
SYMBOL	Defines, displays, and deletes symbols	Usable
TRACE	Displays trace buffer contents	Usable
TRACE_CONDITION	Specifies, displays, and cancels trace conditions for emulation execution	Usable
TRACE_MEMORY	Specifies, displays, and cancels trace data address	Usable
TRACE_MODE	Specifies and displays trace information	Unusable
TRACE_SEARCH	Searches for and displays trace information	Usable

7.2 Emulation Commands

This section provides details of emulation commands in the format shown in figure 7.1.



Command Name

Sect. No.	Command Name Abbreviation	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
- **Abbreviation**
  Abbreviated command name
- **Function**
  Command function
- **Command Format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and suggestions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples.
  Differs a little in the E7000PC.

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 ( ) must be specified.
< >: Contents shown in < > are to be specified or are 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 underlining is often used throughout this manual to indicate input, it is not used in the command format sections 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>: Control register or general-purpose register to be modified or displayed.
(Control register) PC, SR, PR, GBR, VBR, MACH, MACL
(General-purpose register) R0, R1, R2, R3, R4, R5, R6, R7
R8, R9, R10, R11, R12, R13, R14, R15 (SP)
<data>: The value to be set in the specified register

Description

- Modification

— Direct mode

Sets the specified value in the specified register. SP can be specified instead of R15.

```
: .<register> <data> (RET)
```

— Interactive mode

If no data is specified on the command line with the register, register modification is performed in interactive mode. In this case, the system 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), PC, SR, PR, GBR, VBR, MACH, MACL

Example: To modify registers in interactive mode

```
: .<register> (RET)
  <register>          =xxxxxxxx ?         yyyy (RET)
  <register>          =xxxxxxxx ?         yyyy (RET)
     .                     .                    .
     .                     .                    .
```

yyyy <data>: Inputs the value to be set

.: 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 initialization:

R0 to R14	: H'00000000	VBR	: H'00000000
R15 (SP)	: Power-on reset vector value	GBR	: H'00000000
PC	: Power-on reset vector value	MACH	: H'00000000
SR	: H'000000F0	MACL	: H'00000000
PR	: H'00000000		

If the SH-2 is reset by the emulator RESET or CLOCK command, registers are set as follows:

R0 to R14	: The value before reset	VBR	: H'00000000
R15 (SP)	: Power-on reset vector value	GBR	: The value before reset
PC	: Power-on reset vector value	MACH	: The value before reset
SR	: H'000000F0 (I0 to I3 bits are 1)	MACL	: The value before reset
PR	: The value before reset		

Since the reset values for R0 to R14 in the SH-2 are not fixed, the initial values must be set by a program.

Examples

1. To set H'5C60 in the PC, H'FFE00 in the SP, H'FF in R1, and H'11 in R2, and then display all registers:

```
:.PC 5C60  (RET)
:.SP FFE00  (RET)
:.R1 FF  (RET)
:.R2 11  (RET)
:R  (RET)
 PC=00005C60 SR=000003F3:MQIIII-ST
 PR=00000000 GBR=00000000 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-R7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
 R8-R15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 000FFE00
:
```

2. To modify the contents of control registers in interactive mode:

```
:.PC (RET)
 PC  =00001000 ?   2000 (RET)
 SR  =000003F3 : MQIIII--ST ?  303 (RET)
 PR  =00000000 ?   .(RET)
:
```

7.2.2	**!<symbol> or &<symbol> !<symbol> or &<symbol>**	**Displays symbol value**

Command Format

- Display : !<symbol>[ΔD] (RET)
  &<symbol>[ΔD] (RET)

!<symbol>: Symbol to be displayed
&<symbol>: Line number symbol to be displayed
D: Specifies display of data in decimal. (If D is omitted, data is displayed in hexadecimal.)

Description

- Display

Displays data in the format shown in table 7.2, depending on attribute type.

Table 7.2 Display Formats of Symbol Contents

Attribute	Display Format	Description
Function name, structure name, label name, symbol defined with the SYMBOL command	<symbol>: xxxxxxxx	Displays the symbol address as a 4-byte value.
Simple variable name or pointer	<symbol>: xx	Displays the variable contents according to its data length. For decimal display, a signed variable is displayed with a sign.
Array name	<symbol>: xx.. xx.. xx..	Displays entire contents of all elements in the array.
Array element name	<array name> (n): xx .. Indicates the nth element (n = 1, 2, 3, ..)	Displays the contents of the nth element if (n) is specified after array name. Data size to be displayed is the element length of the array.
Line number symbol	<line number symbol> : xxxxxxxx	Displays the start address of a line number if that line number symbol is specified.

Note: A label must be entered from the top of a line with no space, but a space must be entered before an instruction.

Examples

1. To disassemble the contents of an address specified by a symbol:

```
:DA !MAIN_START (RET)
 ADDR     CODE         LABEL          MNEMONIC    OPERAND
 00000500 8800         !MAIN_START
                                      CMP/EQ      #00, R0
 00000502 8905                        BT          00000510
 00000504 5511                        MOV.L       @(4, R1),R5
 00000506 5612                        MOV.L       @(8, R1),R6
 00000508 7601                        ADD         #01, R6
 0000050A E000                        MOV         #00, R0
     .        .                          .              .
     .        .                          .              .
```

2. To display the contents of array msym/sym_chr:

```
:!msym/sym_chr (RET)
  msym/sym_chr:
  00 01 00 02 00 03 00 04 00 05 00 06
:
```

7.2.3	**ABORT** **AB**	**Terminates emulation in parallel mode**

Command Format

- Termination : ABORT (RET)

Description

- Termination

  — Terminates emulation execution in parallel mode (prompt #), and cancels parallel mode.

  — When emulation 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)
  PC=00005C60  SR=000003F3 : MQIIII--ST
  PR=00000000  GBR=00000000   VBR=00000000
  MACH=00000000  MACL=00000000
  R0-R7  00000000 000000FF 00000011 00000000 00000000 00000000 00000000 00000000
  R8-R15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 000FFE00
  +++:BREAK KEY
:
```

Note

When executing the ABORT command in a command chain file, the emulator outputs the reason for stopping (in the same way as when the (BREAK) key or (CTRL) + C keys are pressed), followed by the confirmation message

```
"STOP COMMAND CHAIN (Y/N)".
```

7.2.4	**ASSEMBLE** **A**	**Assembles program**

Command Format

- Line assembly : ASSEMBLE Δ<address> (RET)

  <address>: The address where the object program will be written

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 to create machine codes and is written to memory.

```
: ASSEMBLE  <address>   (RET)
     ADDR         CODE       LABEL            MNEMONIC       OPERAND
   xxxxxxxx       yyyy       <disassemble display>
   xxxxxxxx       yyyy       ?   <subcommand>  (RET)
   xxxxxxxx       yyyy       ?   <subcommand>  (RET)
      (a)          (b)                 (c)
```

(a) Address

(b) Memory contents

(c) Subcommand (Input the contents shown in table 7.3)

The subcommands listed in table 7.3 can be used with the ASSEMBLE command:

Table 7.3 Subcommands for Line Assembly

Subcommand	Description
<assembly language statement>	Assembles the input line (statement) into machine code and writes it to memory.
/[<address1>[∆<address2>]]	Disassembles instructions from <address1> to <address2> and displays them. If <address2> is omitted, the first 16 instructions from <address1> 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 enters subcommand input wait state.
^	Decrements the address (Odd address –1, Even address –2), and enters subcommand input wait state.
.	Terminates the ASSEMBLE command.

Note: The label is entered at the beginning. A space must be entered before an instruction.

— If an undefined label is referenced during line assembly,

```
*** 33:INVALID ASM OPERAND
```

is displayed.

— Even if an odd address is specified, write is performed. In that case, the following warning message is displayed:

```
*** 82:ODD ADDRESS
```

— Line assembly with this command can be performed only between areas CS0 to CS3 and the cache data forced read/write area.

ASSEMBLE

Example

To perform line assembly from address H'1000:

```
:A 1000 (RET)
   ADDR           CODE           LABEL       MNEMONIC    OPERAND
  00001000        0000                       .DATA.W      0000
  00001000        0000  ?        SUB010      MOV         R0, R1 (RET)
  00001002        0000  ?                    ADD         R1, R2 (RET)
  00001004        0000  ?                    JMP         @R3 (RET)
  00001006        0000  ?        . (RET)
  :
```

7.2.5	BREAK B	Sets, displays, and cancels software breakpoints

Command Format

- Setting : BREAKΔ<software breakpoint to be set> [,<software breakpoint to be set>]... (RET)
- Display : BREAK (RET)
- Cancellation : BREAK[Δ]–[<software breakpoint to be cancelled> [,<software breakpoint to be cancelled>]...] (RET)

<software breakpoint to be set>:	<address>[Δ<number of times>]
<address>:	Software breakpoint address
<number of times>:	How many times the specified software breakpoint is to be passed (H'1 to H'3FFF) (Default: H'1)
<software breakpoint to be cancelled>:	The address of the software breakpoint to be cancelled

Note: When the specified address is odd, it is rounded down to an even address.

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 four breakpoints can be set each time this command is issued, and 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 an invalid instruction to cause a break. Do not set software breakpoints at any of the addresses below:

- Addresses specified with the BREAK_SEQUENCE command
- Addresses that hold illegal instructions (H'0000)
- Areas other than CS0 to CS3, and the cache data forced read/write area
- Addresses where BREAK_CONDITION2 settings are satisfied (refer to the following descriptions)
- Addresses containing slot instructions (delayed branch instructions) (refer to the following descriptions)

— The user can also specify the number of times a breakpoint must be reached.

Example: To generate a break when the instruction at address 300 is executed five times

```
BREAK 300 5
```

Note: When multiple passes are specified for a breakpoint, the program must be temporarily stopped 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.

— Software breakpoints are ignored during STEP and STEP_OVER command execution, so the pass count is not updated at this time.

— If a software breakpoint is set at an address located within both the cache and through areas, a break will occur whenever this address is accessed regardless of the area specified.

— Executing instructions set with the BREAK command invalidates BREAK_CONDITION2 settings. Make sure not to set a software breakpoint at an address where a BREAK_CONDITION2 setting is satisfied.

— If a software breakpoint is set at a slot delayed branch instruction, an interrupt caused by a slot illegal instruction occurs without program execution terminating. It is therefore recommended not to set a software breakpoint at a slot delayed branch instruction.

- Display

Display format is as follows:

```
: BREAK  (RET)
   <ADDR>      <CNT>      <PASS>      <SYMBOL>
   xxxxxxxx    yyyy        zzzz         mmmm
     (a)        (b)         (c)          (d)
```

(a) Setting address

(b) Specified number of passes (hexadecimal)

(c) Value of pass counter (shows how many times the specified address has been passed at GO command termination, in hexadecimal)

Note: The pass counter is cleared by the next GO command.

(d) Symbol (only displayed if the software breakpoint address has a symbol)

- Cancellation

  Cancels software breakpoints. Breakpoints can be cancelled in the following two ways:

  — Cancellation of software breakpoints at specified addresses. A maximum of four breakpoints can be cancelled with one command.

  : BREAK–<software breakpoint>[,<software breakpoint>]...(RET)

  — Cancellation of all breakpoints.

  : BREAK– (RET)

Note

In parallel mode, if a memory access command is executed and the emulation stops at a pass point at the same time, the memory access may not take place. In this case,

```
*** 78: EMULATOR POD 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.

Even if the memory contents are modified with the LOAD command, breakpoint settings are not changed.

Examples

1. To set a software breakpoint at H'100:

```
:B 100 (RET)
:
```

2. To generate a break when H'6004 has been passed three times:

```
:B 6004 3 (RET)
:
```

3. To display set software breakpoints:

```
:B (RET)
 <ADDR>          <CNT>     <PASS>    <SYMBOL>
00000100          0001      0000
00006004          0003      0000     !symbol
```

4. To cancel the software breakpoints at H'100:

```
:B - 100 (RET)
:
```

5. To cancel all software breakpoints:

```
:B - (RET)
:
```

7.2.6	BREAK_CONDITION1,2,3,4,5 BC1,2,3,4,5	Specifies, displays, and cancels a hardware break condition

Command Format

- Setting : BREAK_CONDITION (1/2/3/4/5)Δ<condition>[[Δ <condition >]
  [Δ<condition>]...] (RET)
- Display : BREAK_CONDITION [(1/2/3/4/5)] (RET)
- Cancellation : BREAK_CONDITION [(1/2/3/4/5)] [Δ]– (RET)

(1/2/3/4/5): Break command number
When omitted, 1, 2, 3, 4, and 5 will all be displayed or cancelled.
<condition>: Hardware break condition (refer to tables 7.4 to 7.7 for details)

Description

- Setting

— Specifies hardware break conditions. Program execution stops when the specified conditions are satisfied. The specifiable conditions for the five kinds of hardware breaks are summarized in tables 7.4 to 7.7, respectively. BREAK_CONDITION1,2 conditions can also be satisfied in sequential break mode (program execution stops only when BREAK_CONDITION1,2 settings are satisfied in the sequence of condition 2 followed by condition 1). For details on sequential break mode and options, refer to section 7.2.19, GO.

Table 7.4 Specifiable Conditions (BREAK_CONDITION1)

Item and Input Format	Description

Address condition*
A=<address>
PC=<address>[;P]
	The condition is satisfied when the address bus value matches the specified value.
	When A= is selected, the address bus in data access, DMA, or program fetch cycles is specified, and when PC= is selected, the address bus in program fetch cycles is selected.
	When the ;P option is specified with PC=, a break occurs before program execution at the specified address, while if the option is omitted, a break occurs after program execution.
	When PC= is selected, only the delay count specification condition is valid.

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 long words, respectively. Note that when a data condition is specified, access cannot be performed in program fetch cycles.

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).
Access type DAT: Execution cycle DMA: DMA cycle Default: All bus cycles described above (including program fetch cycle)	The condition is satisfied when the bus-cycle type matches the specified type. Multiple access types cannot be specified; either select one of the access types on the left, or specify none of them.
Delay count specification DELAY=<value> <value>: H'1 to H'FFF	This condition can be specified in combination with any of the above conditions. The complete condition combination is satisfied when the specified number of bus cycles has been executed after the other specified condition is satisfied.

Note: * Refer to the address and data condition descriptions on the following pages.

Table 7.5 Specifiable Conditions (BREAK_CONDITION2)

Item and Input Format	Description

Address condition*
PC=<address>[;P]
A=<address>
	The condition is satisfied when the address bus value matches the specified value.
	When A= is selected, the address bus in data access, DMA, or program fetch cycles is specified, and when PC= is selected, the address bus in program fetch cycles is selected.
	When the ;P option is specified with PC=, a break occurs before program execution at the specified address, while if the option is omitted, a break occurs after program execution.
	When PC= is selected, other conditions cannot be specified.

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).
Access type DAT: Execution cycle DMA: DMA cycle Default: All bus cycles described above (including program fetch cycle)	The condition is satisfied when the bus-cycle type matches the specified type. Multiple access types cannot be specified; either select one of the access types on the left, or specify none of them.

Note: * Refer to the address condition descriptions on the following pages.

Table 7.6 Specifiable Conditions (BREAK_CONDITION3,4)

Item and Input Format	Description
Address condition A=<address>	The condition is satisfied when the address bus value matches the specified value.
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).

Table 7.7 Specifiable Conditions (BREAK_CONDITION5)

Item and Input Format	Description
Probe condition PRB	The condition is satisfied when a trigger is input from the probe pin.

- The data conditions of BREAK_CONDITION1 are satisfied when the address bus and data bus values match the specified values. The data bus (the SH-2 internal bus) is always 32 bits long. Note the following when specifying break conditions.

  - Long-word access

    Long-word data is accessed in one bus cycle. Only long-word 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 can be used when specifying BREAK_CONDITION1,2 address, PC, and data conditions. When a bit is masked, the condition is satisfied irrespective of its bit value. To mask a bit, specify it as * at input. Examples of masks are shown below and in table 7.8.

  Example 1: The following condition is satisfied when the four low-order bits of the address condition are not specified and the D0 bit is 0 in the byte data condition.

  ```
  : BREAK_CONDITION1 A=H'400000* D=B'*******0
  ```

  Example 2: The following condition is satisfied when bits 8 to 11 are 2 (the other bits are not specified) in the PC condition.

  ```
  : BREAK_CONDITION2 PC=H'*****2**
  ```

Table 7.8 Mask Specifications (BREAK_CONDITION1,2)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	D0 and D5 bits	Address, PC, and data (D, WD, or LD)
Hexadecimal	4 bits	H'F**50	D15 to D8 bits	Address, PC, and data (D, WD, or LD)

- If a hardware break condition is specified and that condition is satisfied, emulation stops after two or more instructions have been executed.

- Display

Displays specified conditions. The specified input character string is displayed as is before. All five break conditions are displayed if break command numbers 1 to 5 are omitted. For BREAK_CONDITION1, delay count since the previous break condition was satisfied is displayed. If no condition is specified, a blank is displayed.

Note that delay count after conditions are satisfied will not be displayed correctly unless trace display has been performed at least once.

```
: BC (RET)
  BC1 (BREAK_CONDITION1 command setting)
  DELAY COUNT = xxxx                          xxxx:  Delay count after the condition is satisfied
  BC2 (BREAK_CONDITION2 command setting)
  BC3 (BREAK_CONDITION3 command setting)
  BC4 (BREAK_CONDITION4 command setting)
  BC5 (BREAK_CONDITION5 command setting)
```

- Cancellation

Cancels specified conditions. When conditions 1, 2, 3, 4, and 5 are omitted, all break conditions are cancelled.

— Cancels all break conditions

```
: BREAK_CONDITION – (RET)
```

— Cancels BREAK_CONDITION2

```
: BREAK_CONDITION2 – (RET)
```

Notes

1. BREAK_CONDITION2 is ignored when a stop address is specified with the GO command or during STEP and STEP_OVER command execution.

2. When specifying an address condition, all address bits are checked. Therefore, if a break is set at an address from the cache area, a break will not occur when that address is accessed from the through area which occupies the same memory space. To generate a break at both the cache and through areas, mask the address specifications set with the BC1 or BC2 commands.

3. Executing addresses containing software breakpoints (set by the BREAK and BREAK_SEQUENCE command) invalidates the BREAK_CONDITION2 settings. Make sure not to set software breakpoints at addresses where BREAK_CONDITION2 settings are satisfied.

4. A slot delayed branch instruction cannot terminate user program execution before a PC break occurs; setting an execution stop condition in a slot delayed branch instruction will not stop emulation before PC break execution.

5. The bus-cycle condition is satisfied when PC relative memory (MOV.W @(10,PC),R0 and so on) is accessed in execution cycles (DAT is specified as the access type). Note that this specification differs from the TRACE_CONDITION and TRACE_SEARCH commands.

6. The BREAK_CONDITION1,2 settings are implemented by the SH-2 user break controller. Accordingly, the user cannot use the SH-2 user break controller while the emulator is being used.

Examples

1. To generate a break when byte data H'10 is accessed at address H'F000000:

```
:BC1 A=F000000 D=10  (RET)
:
```

2. To generate a break when data is written to address H'1000000:

```
:BC2 A=1000000 W DAT  (RET)
:
```

3. To display the specified condition:

```
:BC  (RET)
 BC1 A=F000000 D=10
 DELAY COUNT=0000
 BC2 A=1000000 W DAT
 BC3
 BC4
 BC5
:
```

4. To delete the specified condition:

```
:BC1 -  (RET)
:BC2 -  (RET)
:
```

7.2.7	**BREAK_SEQUENCE** **BS**	**Sets, displays, or cancels software breakpoints with pass sequence specification**

Command Format

- Setting : BREAK_SEQUENCEΔ<pass point>Δ<pass point>[Δ<pass point>
  [Δ<pass point>]] (RET) (Pass point setting)
  : BREAK_SEQUENCEΔ<reset point>;R (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>:	<address> (two to four points)
R:	Reset point specification
<reset point>:	<address> (one point)

Note: The <address> is rounded down to an even value, if the specified address is odd.

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.

  — 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.

— Do not set a pass point or a reset point at any of the addresses below:

- Addresses specified with the BREAK command
- Addresses that hold illegal instructions (H'0000)
- Areas other than CS0 to CS3, and the cache data forced read/write area
- Addresses where BREAK_CONDITION2 settings are satisfied (refer to the following descriptions)
- Addresses containing slot instructions (delayed branch instructions) (refer to the following descriptions)

— 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.

— Executing instructions set with the BREAK_SEQUENCE command invalidates BREAK_CONDITION2 settings. Make sure not to set a software breakpoint at an address where a BREAK_CONDITION2 setting is satisfied.

— If a software breakpoint is set at a slot delayed branch instruction, an interrupt caused by a slot illegal instruction occurs without program execution terminating. It is therefore recommended not to set a software breakpoint at a slot delayed branch instruction.

- Display

Displays specified pass points and reset point as follows:

```
: BREAK_SEQUENCE  (RET)
 PASS POINT NO.1    = xxxxxxxx    yyyy    zzzzzzzz
 PASS POINT NO.2    = xxxxxxxx    yyyy    zzzzzzzz
 PASS POINT NO.3    = xxxxxxxx    yyyy    zzzzzzzz
 PASS POINT NO.4    = xxxxxxxx    yyyy    zzzzzzzz
 RESET POINT        = xxxxxxxx    yyyy    zzzzzzzz
                         (a)       (b)      (c)
```

(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'FFFF, counting restarts from H'0. The number of times passed is cleared by the next GO command.)

(c) Symbol name (Displayed only when specified with symbols.)

- Cancellation

  Cancels specified pass points or reset point.

  — Cancellation of pass points

  ```
  : BREAK_SEQUENCE– (RET)
  ```

  — Cancellation of a reset point

  ```
  : BREAK_SEQUENCE–;R (RET)
  ```

Note

In parallel mode, if a memory access command is executed and the emulation stops at a pass point at the same time, the memory access may not take place. In this case,

```
*** 78: EMULATOR POD 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.

Examples

1. To set pass points at H'4000, H'4100, H'4200, and H'4300 in that order and a reset point at 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
 PASS POINT NO2 = 00004100   0000
 PASS POINT NO3 = 00004200   0000
 PASS POINT NO4 = 00004300   0000
 RESET POINT    = 00002000   0000
:
```

3. To cancel the reset point:

```
:BS - ;R  (RET)
:
```

4. To cancel the pass points and reset point:

```
:BS -  (RET)
:BS - ;R  (RET)
:
```

7.2.8	**CHECK** **CH**	**Tests SH-2 pins**

Command Format

- Test : CHECK (RET)

Description

- Test

Tests the status of the SH-2 pins shown in table 7.9.

Table 7.9 SH-2 Pin Test

Pin Name	Error Status	Remarks
RES	RES 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.	

If an error occurs,

FAILED AT <pin name>

is displayed.

Example

When the IRL0 signal is low:

```
:CH  (RET)
 FAILED AT IRL0
:
```

7.2.9 CLOCK CL — Sets or displays clock

Command Format

- Setting : CLOCKΔ<clock> (RET)
- Display : CLOCK (RET)

<clock>: One of the following signals:
E: Emulator internal clock signal (6.144 MHz)
(Only available when one of the SH-2 clock modes 0 to 3 is selected)
U: User system clock

Description

- Setting

— When one of the clock modes 0 to 3 is specified, selects emulator clock signals from the user system or from the emulator clock (installed in the emulator). Resets the SH-2 when a clock is selected, and consequently, internal I/O registers and control registers return to their reset values.

— Displays the specified clock signal. If the user system clock (U) is specified, but the clock signal is not input, an error occurs and the emulator clock (E) is set instead (when the clock mode is 0 to 3). At emulator initiation, the user system and emulator clocks are selected in that order, and the correct clock signal is set. When the clock mode is specified to be among 4 to 6 by the MODE command, the emulator clock cannot be selected. Confirm that a clock signal is entered from the user system before using the emulator.

- Display

Displays the current clock signal.

```
: CLOCK  (RET)
  CLOCK = <Used clock>
```

<Used clock>: EML: Emulator clock
USER: User system clock

Note

If U is specified and the following clock signal problem occurs, the emulator system program may terminate. In this case, the emulator system program must be 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:

```
:CL U (RET)
 ** RESET IN BY E7000 !
 CLOCK = USER
:
```

2. To use the emulator clock signal:

```
:CL E (RET)
 ** RESET IN BY E7000 !
 CLOCK = EML
:
```

3. To display the current clock signal:

```
:CL (RET)
 CLOCK = EML
:
```

7.2.10	**COMMAND_CHAIN** **CC**	**Inputs emulator command from a file** **(specific to the E7000)**

Command Format

- Command input   : COMMAND_CHAIN∆<file name> (RET)

Description

- Command input

— Sequentially reads commands from a command file, and executes them.

When the following command file is specified, MAP, MEMORY, and CLOCK command are sequentially executed. The MEMORY command, though requiring further input within the command, can be read from a file and be executed. However, this command cannot execute the COMMAND_CHAIN command itself.

Example:

```
File contents:  MAP    1000000   101FFFF;S
                MEMORY   100
                30
                .
                CLOCK
```

```
Execution results:  : COMMAND_CHAIN   <file name>  (RET)
                    : MAP    1000000    101FFFF;S
                      REMAINS EMULATION MEMORY     S=D'0384kB
                    : MEMORY        100
                      00000100        00            ?   30
                      00000101        00            ?   .
                    : CLOCK
                      CLOCK  =  USER
                    : (Command input wait state)
```

- The command file reading does not terminate until the end of the file is detected, or the (BREAK) key or (CTRL) + C keys are pressed. If either combination of keys are pressed, the message below is displayed, and execution is halted. The COMMAND_CHAIN command is then continued or terminated.

  STOP COMMAND CHAIN (Y/N) ? (a) (RET)

  (a) Y: Terminate
  N: Continue

- Create a command file with the host system editor connected to the emulator and transmit it to an emulator file using the TRANSFER, INTFC_TRANSFER, or LAN_TRANSFER command.

- This command is specific to the E7000. This command cannot be used in the E7000PC. The E7000PC, however, supports a function to input commands from the IBM PC file automatically. For details, refer to section 3.7.2, Debugging Function, in Part II, E7000 Guide.

Example

To execute command file SAMPLE.COM:

```
:CC SAMPLE. COM  (RET)
:FILL 0 FFFF
:MEMORY 100
      :
```

The command is input sequentially and then executed.

7.2.11	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, and ASCII format. Input data is handled as 4-byte values, but unnecessary (leading) zeros are not displayed. If there is no corresponding ASCII character, a period (.) is displayed instead.

```
: CONVERT  <data>   (RET)
    H'xx...      D'xxx...    Q'xxx...      B'xxx...      xx...
      (a)          (b)         (c)           (d)          (e)
```

(a) Hexadecimal display
(b) Decimal display
(c) Octal display
(d) Binary display
(e) ASCII display

— If the H', D', Q', or B' radix is not specified for <data> at data input, the radix specified with the RADIX command is assumed.

Examples

1. To convert hexadecimal data (H'7F):

```
:CV H'7F  (RET)
 H'7F D'127 Q'177 B'1111111   ....
:
```

2. To convert the formula:

```
:CV H'31+D'16  (RET)
 H'41  D'65 Q'101 B'1000001   ...A
:
```

7.2.12	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>][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
Y: Specify Y if a confirmation message is not necessary. If Y is specified, data in all assigned areas is replaced without confirmation messages.

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 not specified, the following message is displayed when 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; search for another occurrence of the specified data continues. To terminate this command before reaching <end address>, press the (CTRL) + C keys.

If option Y is specified, data is replaced without confirmation messages.

— If <data 1> is not found at any point in the replacement range,

```
*** 45:NOT FOUND
```

is displayed.

— Memory modification with this command can be performed only between areas CS0 to CS3 and the cache data forced read/write area.

Examples

1. To replace 2-byte data H'6475 in the area 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 area from H'FB80 to H'FE00 with 'DATE' (without confirmation message):

```
:DC 'DATA' 'DATE' FB80 FE00 ;L Y  (RET)
:
```

7.2.13	DATA_SEARCH DS	Searches for memory data

Command Format

- Search : DATA_SEARCHΔ<data>[Δ<start address>
  [(Δ<end address>/Δ@<number of bytes>)]][;[<size>] [ΔN]] (RET)

<data>:	Data to be searched for
<start address>:	Search-start address (Default: H'0)
<end address>:	Search-end address (Default: Maximum address: H'FFFFFFFF)
<number of bytes>:	The number of bytes to be searched for (Default: Maximum address in each operating mode)
<size>:	Length of data to be searched for B: 1 byte W: 2 bytes L: 4 bytes Default: 1 byte
N:	Data other than the specified data is searched for

Description

- Search

— Searches for <data> from the start address to the end address. All addresses where <data> is found are displayed.

— If data is not found,

```
*** 45:NOT FOUND
```

is displayed.

— If the N option is specified, data other than the specified <data> is searched for.

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 addresses starting from the H'1000:

```
:DS 0 1000 @100 ; W N (RET)
 *** 45 : NOT FOUND
:
```

7.2.14	DISASSEMBLE DA	Disassembles and displays memory contents

Command Format

- Display : DISASSEMBLEΔ<start address> [(Δ<end address>/Δ@<number of instructions>)] (RET)

<start address>:	Start address of disassembly
<end address>:	End address of disassembly
<number of instructions>:	The number of instructions to be disassembled

Description

- Display

— Disassembles the specified memory contents and displays addresses, machine codes, labels, mnemonics, and operands in the following format.

As many lines as necessary are used for the display.

```
  ADDR          CODE           LABEL    MNEMONIC     OPERAND
<address>   <machine code>    <label>  <mnemonic>   <operand>
```

— If <end address> or <number of instructions> is omitted, 16 lines of data are disassembled and displayed.

— If there is no applicable instruction,

```
DATA.W   xxxx
```

is displayed.

If <start address> is odd,

```
DATA.B   xx
```

is displayed.

— After executing this command (except when it is forcibly terminated by the (CTRL) + C keys or (BREAK) key, or by an error), press the (RET) key to disassemble and display the next 16 lines of data.

— Disassemble can be performed only in areas CS0 to CS3 and in the cache data forced read/write area.

Examples

1. To disassemble and display six instructions starting from address H'1000:

```
:DA  1000  @6  (RET)
   ADDR   CODE      LABEL    MNEMONIC    OPERAND
  00001000 E000     !MAIN     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 16 instructions again by only entering (RET):

```
:DA  1000    (RET)
   ADDR   CODE      LABEL    MNEMONIC    OPERAND
 00001000  1F01     !SUB      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      LABEL    MNEMONIC    OPERAND
 00001020  2100               MOV.B       R0,@R1
 00001022  2201               MOV.W       R0,@R2
 00001024  2303               MOV.L       R0,@R3
  :                         :
  :                         :
```

7.2.15	DUMP D	Displays memory contents

Command Format

- Display : DUMPΔ<start address>[(Δ<end address/Δ[@]<number of bytes>)]
  [;<display unit>] (RET)

<start address>: Display start address

<end address>: End address for memory dump

<number of bytes>: Size of data for memory dump
If @ is omitted, this value is taken as end address or size according to the inequalities given below. Default is 256 bytes, as size.
End address: <start address> ≤ specified value
Size: <start address> > specified value

<display unit>: Size in bytes of display unit
B: 1 byte
W: 2 bytes
L: 4 bytes
Default: 1 byte

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.

- If (CTRL) + P keys (hold down (CTRL), then press P) are entered during a memory dump, the emulator displays the 256 bytes of data before the start address of the current dump, and halts command execution.

  The emulator then waits for key input, but does not display a prompt. If the (RET) key is pressed at this stage, the display scrolls through the memory contents until the specified end address is reached. If instead, (CTRL)+ P keys are pressed, the 256 bytes of data before the start address of the last dump are displayed.

- If no command is executed after DUMP command execution has been terminated (except for forcible termination), the 256 bytes of data from the next address of the last dump are displayed.

Note

This command can be executed in parallel mode; however, emulation does not operate in realtime. Refer to section 1.3.3, Parallel Mode, for more information.

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 37 30 30 30  "           E7000"
:
```

2. To display 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    "................"
:
```

3. To display by entering (CTRL) + P and (RET):

```
:D 1000 (RET)
 <ADDRESS>             <    D   A    T   A    >                  <ASCII CODE>
 00001000  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001010  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
                      Enter (CTRL) + P
 00000F00  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00000F10  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00000F20  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00000F30  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
    :               :                    :                     :
 00000FF0  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
      Display of memory dump stops.     Enter (RET) to continue display.
 00001000  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001010  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001020  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001030  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
    :               :                    :                     :
 000010F0  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
:(RET)               Entering (RET) displays next 16 lines.
 00001100  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001110  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001120  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001130  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
    :               :                    :                     :
 000011F0  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
:
```

7.2.16	**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:

```
:GO (RET)
 ** PC=00003400        (RET)          (Parallel mode entered)
#M FD80 (RET)
 0000FD80  00    ?  FF(RET)           (Command execution in parallel mode)
 0000FD81  10    ?  .(RET)
#END (RET)                            (Parallel mode cancellation)
 ** PC=00003800
```

7.2.17	EXECUTION_MODE EM	Specifies and displays execution mode

Command Format

- Setting : EXECUTION_MODE [ΔRES=<option 1>][ΔBRQ=<option 2>]
  [ΔTIME=<option 3>][ΔBS=<option 4>][ΔTRG=<option 5>]
  [ΔREAL=<option 6>][ΔCT=<option 7>][ΔMON=<option 8>] [;C] (RET)
- Display : EXECUTION_MODE (RET)

<option 1>: Specifies whether the reset signal is output from the WDTOVF pin of the SH-2 to the user system at cycle reset or clock-switching reset, or when the RESET command is executed.

E: Outputs the reset signal (System reset is output from the WDTOVF pin; default at emulator shipment)

D: Does not output the reset signal (System reset is not output from the WDTOVF pin)

<option 2>: Specifies whether the BREQ (bus request) signal inputs are enabled.

E: Enables the BREQ signal input (default at emulator shipment)

M: Disables the BREQ signal inputs during emulator operation and enables the signal input during program execution

D: Disables the BREQ signal inputs

<option 3>: Specifies the minimum time to be measured by the GO command execution.

1: 1 µs (default at emulator shipment)

2: 250 µs

<option 4>: Displays expanded functions.

<option 5>: When hardware break conditions (set by the BREAK_CONDITION1,2 command) are satisfied, specifies whether a pulse is output from the trigger output pin of the pod without a break. (The BREAK_CONDITION3,4,5 command does not have this function.)

E: Outputs a trigger without a break

M: Break occurs and outputs a trigger

D: Break occurs but does not output a trigger (default at emulator shipment)

<option 6>: Specifies memory access mode.

E: Realtime mode

D: Gate insertion mode (default at emulator shipment)

<option 7>: Specifies whether the cache access trace is enabled.

E: Enables the cache access trace (default at emulator shipment)

D: Disables the cache access trace

<option 8>: Specifies execution status display time interval.

0: No display
1: Approximately 200 ms (default at emulator shipment)
2: Approximately 2 s

C: Writes the setting contents to the configuration file.

Description

- Specification

— Enables/disables the reset signal output from the WDTOVF pin of the SH-2 to the user system at cycle reset or clock-switching reset, or when the RESET command is executed. Setting this bit has no effect on WDT.

- To disable the reset signal output from the WDTOVF pin of the SH-2:

```
: EXECUTION_MODE RES=D (RET)
```

- To enable the reset signal output from the WDTOVF pin of the SH-2:

```
: EXECUTION_MODE RES=E (RET)
```

— Enables/disables the BREQ signal (bus request signals) inputs during realtime emulation. In the emulator used to debug the slave CPU, the BREQ signals are always input regardless of this option setting.

- To disable the BREQ signal inputs during emulator operation:

```
: EXECUTION_MODE BRQ=D (RET)
```

- To disable the BREQ signal inputs during emulator operation but enable the BREQ signal input during program execution:

```
: EXECUTION_MODE BRQ=M (RET)
```

- To enable the BREQ signal inputs during emulator operation and program execution:

```
: EXECUTION_MODE BRQ=E (RET)
```

By setting this option, the emulator can debug the user system under the condition summarized in table 7.10. When the E7000 is used for debugging the user system where two SH-2s (master and slave) are used, disabling the BREQ signal inputs by setting BRQ = D or M allows the debugging of the master CPU without operating the slave CPU.

Table 7.10 BREQ Signal Inputs in Master/Slave CPUs

Option Setting	BREQ Inputs at Emulator Connected to Master CPU		BREQ Inputs at Emulator Connected to Slave CPU	
	Emulator Operation	Program Execution	Emulator Operation	Program Execution
BREQ = E	Enabled	Enabled	Enabled	Enabled
BREQ = M	Disabled	Enabled	Enabled	Enabled
BREQ = D	Disabled	Disabled	Enabled	Enabled

— Specifies the minimum time to be measured by the GO command execution.

- To set the minimum time to 1 µs:

  : EXECUTION_MODE TIME=1 (RET)

- To set the minimum time to 250 ns:

  : EXECUTION_MODE TIME=2 (RET)

— Specifies whether or not to output a pulse and continue program execution when hardware break conditions set by the BREAK_CONDITION1,2 command are satisfied. When a hardware break condition is satisfied, a pulse is output from the trigger output pin. However, when hardware break conditions set by the BREAK_CONDITION3,4,5 command are satisfied, a break occurs regardless of this specification. Refer to section 1.7, Trigger Output, for details.

- To terminate program execution when hardware break conditions are satisfied:

  : EXECUTION_MODE TRG=D (RET)

- To terminate program execution and output a pulse when hardware break conditions are satisfied:

  : EXECUTION_MODE TRG=M (RET)

- To output a pulse and continue program execution when hardware break conditions are satisfied:

  : EXECUTION_MODE TRG=E (RET)

— Displays the expanded functions. Enter numerical or alphanumerical values from H'0 to H'7C.

— Selects realtime mode or gate insertion mode. Refer to section 1.3, Realtime Emulation, for details of modes.

- To select gate insertion mode:

  : EXECUTION_MODE REAL=D (RET)

- To select realtime mode:

  : EXECUTION_MODE REAL=E (RET)

— Specifies whether or not trace information is acquired during cache access. Acquiring trace information during cache accesses degrades the SH-2 operating speed because the cache trace function uses the external bus. In a user system where two SH-2s (master and slave) are used, the operating speed of the SH-2 that is not acquiring trace information is significantly degraded.

- To enable trace information acquisition during cache access:

  : EXECUTION_MODE CT=E (RET)

- To disable trace information acquisition during cache access:

  : EXECUTION_MODE CT=D (RET)

— Specifies PC display time interval during GO command execution:

- Not to 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)

— When the C option is specified, the following message is displayed:

CONFIGURATION WRITE OK (Y/N)?

In the E7000, writes the settings to the configuration file in the system disk, when Y is input. In the E7000PC, writes the settings to the IBM PC configuration file in the same directory as the system disk, when Y is input. The same settings will be valid after the emulator is activated with this system disk.

- Display

When all options are omitted, the current value is 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, modification before entering a period is valid.

```
: EXECUTION_MODE (RET)
  RES=E  BRQ=D  TIME=1uSEC  BS=02  TRG=D  REAL=E  CT=E  MON=1        (Displays current value)
  RES (E: ENABLE/D: DISABLE) ?  D (RET)              (Does not output reset)
  BRQ (E: ENABLE/D: DISABLE) ?  (RET)                (No modification)
  TIME (1: 1uSEC/2: 0.25uSEC) ?  . (RET)             (Command is terminated. Modification is valid.)
:
```

Notes

1. In EXECUTION_MODE command description, SH-2’s BRLS/BACK input and BGR/BREQ output are referred to as BREQ (bus request signals) inputs.

2. If cache access trace is disabled by the CT option, instruction display with TRACE command is invalid. In addition, the user program is executed only in the cache, the PC during user program execution cannot be displayed. For details, refer to section 7.2.19, GO.

3. The BRQ and CT options may not be modified because the SH-2’s bus mastership cannot be obtained. If it is so, other options are modified.

Examples

1. To specify the realtime mode and write the settings to the configuration file:

```
:EM REAL=E ;C (RET)
 CONFIGURATION WRITE OK(Y/N)?
:
```

2. To display the specified values of the current emulation execution mode and modify them in interactive mode (the command execution can be terminated by entering a period (.)):

```
:EM (RET)
 RES=E  BRQ=D  TIME=1uSEC  BS=02  TRG=D  REAL=E  CT=E
 RES (E:ENABLE/D:DISABLE) ?  (RET)                 (Input (RET) for no modification)
 BRQ (E:ENABLE/D:DISABLE) ?  (RET)
 TIME (1:1uSEC/2:0.25uSEC) ?  2 (RET)
 BS (0 - 7C) ?  .(RET)             (Command is terminated and new setting becomes valid)
:
```

7.2.18 FILL F — Writes data to memory

Command Format

- Write: FILLΔ<start address>(Δ<end address>/Δ@<number of bytes>)[Δ<data>] [;[<size>][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
N: No verification

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)

— Data can be written to only areas CS0 to CS3 and the cache data forced read/write area.

— 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 a size, the lower bits are rounded down to become a multiple of four. Writing never exceeds the specified <end address>.

FILL

Example

To fill the entire area from addresses H'0 to H'6FFF with 1-byte data H'00:

```
:F 0  6FFF 0  (RET)
:
```

7.2.19	GO G	Provides realtime emulation

Command Format

- Execution : GO[Δ[<start address>][;[<break address>][Δ<mode>][ΔTM]]] (RET)

<start address>: Start address of realtime emulation, or the word RESET

<break address>: A breakpoint address (A break occurs before an instruction at break address is executed)

<mode>: Emulation mode

SB: Sequential break mode

R=<n>: Cycle reset mode, n = 1 to 9

N: Temporarily invalidates break conditions

TM: Displays the memory contents in the address specified with the TM (TRACE_MEMORY) command.

Description

- Execution

— Executes realtime emulation (user program execution) starting with a specified <start address>. The following data can be specified as <start address>.

: GO RESET (RET) : After RES signal input to the SH-2, PC and SP are set to the values specified with the reset vector and program execution starts.

: 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.

— By the <mode> specification at the GO command input, the program executes in one of the following modes.

- Sequential break mode (SB)

  Realtime emulation stops only when break conditions set by the BREAK_ CONDITION1,2 command are satisfied in the sequence of <condition 2> followed by <condition 1>.

- Cycle reset mode (R=n)

  A RES signal is input to the SH-2 at the intervals given in table 7.11. At the same time, a trigger signal for an oscilloscope is output through the trigger output probe pin. In this mode, all break conditions and trace conditions are invalidated during emulation.

- Temporary invalidation of break conditions

  If the N option is specified, software breakpoints set with the BREAK or BREAK_SEQUENCE command and hardware breakpoints set with the BREAK_CONDITION1,2,3,4,5 command are invalidated temporarily, and emulation continues. However, the breakpoints are invalidated only within one GO command emulation. If option N is not specified in the next GO command emulation, breakpoints are validated again.

Table 7.11 Cycle Reset Times

Value of n	**Reset Interval**
1	32 µs
2	96 µs
3	512 µs
4	1.024 ms
5	5.12 ms
6	10.24 ms
7	51.2 ms
8	102.4 ms
9	512 ms

Table 7.12 lists restrictions for the above modes.

Table 7.12 Restrictions for Realtime Emulation Modes

Mode	Restrictions
Sequential break mode	• Conditions must be specified with the BREAK_CONDITION1,2 command.
Cycle reset mode	• Software break conditions specified with the BREAK or BREAK_SEQUENCE command are ignored. • Hardware break conditions specified with the BREAK_CONDITION1,2,3,4,5 command are ignored. • All conditions specified with the TRACE_CONDITION command are ignored. • Parallel mode cannot be entered. • The TM option cannot be specified.

— If <break address> is specified, realtime emulation stops after the instruction at the specified address is executed. This specification is valid for only the current GO command emulation. BREAK_CONDITION2 is invalid when a break address is specified.

— Program fetch addresses are displayed every 200 ms during realtime emulation. If the TM option is specified, memory contents of the address specified with the TRACE_MEMORY command are also displayed. One-byte data is displayed for memory contents. An error occurs if the address is not specified with the TRACE_MEMORY command.

```
: GO ; TM  (RET)
 ** PC = xxxxxxxx      yyyyyyyy = zz
```

xxxxxxxx:	Program fetch address
yyyyyyyy:	Address of memory contents to be displayed (valid when the TM option is specified)
zz:	Memory contents (1 byte)

Note: The TRACE_MEMORY display address can be modified in parallel mode, but the contents to be displayed are not updated until access is completed. (When specifying access with DMA, the display contents are not updated until access is completed with DMAC: the contents before modification are displayed.) Display contents of PC relative memory access (MOV.W @(10,PC),R0 and so on) are not updated.

- During GO command emulation, pressing the SPACE key or (RET) key sets parallel mode. For details, refer to section 1.3.3, Parallel Mode.

- If emulation is terminated, register contents, execution times, cause of termination, and line number symbols are displayed in the following format:

```
PC=00001000  SR=000003F3:MQIIII--ST                                     (a)
PR=00000000  GBR=00000000  VBR=00000000
MACH=00000000  MACL=00000000
R0 – 7    00000000  00000001  00000002  00000003  00000004  00000005  00000006
R8 – 15  00000008  00000009  00000000  00000000  00000001  00000002  00000003
RUN-TIME=D'000H:00M:00S:000000US[:000NS]                                (b)
+++: <cause of termination>                                             (c)
LINE NO = <line number symbol> + n                                      (d)
```

(a) The contents of each register at emulation termination.

(b) Time of user program execution, in decimal. According to the TIME option of the EXECUTION_MODE command, the maximum measurable time is 305 or 76 hours, where the minimum measurement time is 1 µs or 250 ns, respectively. If the period exceeds the maximum, it is displayed as *.

(c) Cause of termination, as listed in table 7.13.

(d) If a line number symbol is defined, the termination location is displayed in the format of: line number symbol + n.

Table 7.13 Causes of GO Command Emulation Termination

Display	Termination Cause
BREAK KEY	The (CTRL) + C keys were pressed or the ABORT command was executed for forcible termination
BREAK POINT	Emulation stopped at a breakpoint specified with the BREAK command
STOP ADDRESS	An instruction at the break address was executed
BREAK SEQUENCE	Software break condition specified with the BREAK_SEQUENCE command was satisfied
BREAK CONDITION 1	A break condition specified with the BREAK_CONDITION1 command was satisfied
BREAK CONDITION 2	A break condition specified with the BREAK_CONDITION2 command was satisfied
BREAK CONDITION 3	A break condition specified with the BREAK_CONDITION3 command was satisfied
BREAK CONDITION 4	A break condition specified with the BREAK_CONDITION4 command was satisfied
BREAK CONDITION 5	A break condition specified with the BREAK_CONDITION5 command was satisfied
BREAK CONDITION 1, 2,3,4,5	A break condition was satisfied when the break conditions specified with the BREAK_CONDITION1,2,3,4,5 command were satisfied (refer to notes)
BREAK CONDITION SB	Sequential break conditions specified with the BREAK_CONDITION1,2 command were satisfied
GUARDED AREA ACCESSED	A guarded memory area was accessed
WRITE PROTECT	A write-protected area was written to
ILLEGAL INSTRUCTION	A break instruction (H'0000) was executed
NO EXECUTION	The user program was not executed (this message is displayed only for the RESULT command)
RESET IN BY E7000	Terminates the program with the RES signal because an error occurs in the user system
DMA GUARDED OR WRITE PROTECT	The write-protected area is written to or guarded memory area is accessed by the DMA during the continuous execution after the software breakpoint

— During user program execution, SH-2 execution status is displayed. Displayed contents are shown in table 7.14. This status is monitored every 200 ms and if there is a difference from the previous status, the status is displayed.

Table 7.14 Execution Status Display

Display	Meaning
** RUNNING*[1]	The user program execution is initiated. This message is displayed once when GO command execution is started or when parallel mode is cancelled. Note that this message will be deleted when **PC=xxxxxxxx [yyyyyy=zz] is displayed.
**PC=xxxxxxxx [xxxxxxxx = xxxxxxxx] (a) (b) (c) (a) Program fetch address (b) Memory address (c) Memory contents	Current program fetch address is displayed. When specifying TM option with GO command, the memory address and its contents are displayed.
** VCC DOWN	User system Vcc (power voltage) is 4 V or less.
** RESET	RES signal is low. The SH-2 has been reset.
** WAIT A = xxxxxxxx xxxxxxxx: Address bus value	WAIT signal is low.
** BREQ*[2]	BREQ signal is low.
** TOUT A = xxxxxxxx*[2] xxxxxxxx: Address bus value	The address bus value is displayed. The bus cycle stops for 80 µs or more.
** PREQ	Clock pause signal is low.

Notes: 1. **PC=xxxxxxxx [yyyyyy=zz] will not be displayed if the cache access trace is disabled by specifying D in the CT option of the EXECUTION_MODE command or if the user program is executed only in the cache.

2. These messages are not displayed if the cache access trace is disabled.

Notes

1. When the hardware break condition (BREAK CONDITION1,2,3,4,5 command setting) is satisfied during program execution, the program does not terminate until at least one of the instructions that have been already fetched is executed. If another hardware break is satisfied before the user program terminates, BREAK CONDITION1,2,3,4,5 (the number of the satisfied condition) is displayed. For further details, examine the trace information.

2. At each software breakpoint set with the BREAK command or the BREAK_SEQUENCE command, the program stops, the pass count and address of the program are analyzed, and then the program continues. When the memory command processing in parallel mode occurs during this termination, memory cannot be accessed. At this time,

```
*** 78: EMULATOR POD BUSY
```

is displayed, and the command should be re-input. However, when the interval of termination is short, the PC is not displayed, the emulator does not enter parallel mode, or parallel mode command may not be executed.

3. When the contents of a breakpoint (set with the BREAK or BREAK_SEQUENCE command, or a stop address) have been modified by the user program during emulation, that breakpoint will be cancelled at execution stop.

Examples

1. To reset the SH-2 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 when address H'2020 is executed:

```
:G  1000 ; 2020 (RET)
 ** PC=00002002
```

3. To start emulation from the current PC address in the sequential break mode:

```
:G ; SB (RET)
 ** PC=00004250
```

4. To invalidate current software break conditions and hardware break conditions, start emulation, and display the contents of the address specified by the TRACE_MEMORY command:

```
:G ; N TM (RET)
 ** PC=00006642    0000FFD0=80
```

5. 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
```

7.2.20	**HELP** **HE**	**Displays all commands and command format**

Command Format

- Display : HELP (RET) (All commands are displayed.)
  : HELP Δ <command> (RET) (Command format is displayed.)

Description

- Display:

  — Displays all emulator command names and abbreviations.

Examples

1. To display all emulator commands:

```
:HELP (RET)
    .<REGISTER>                               *!<SYMBOL>
   *&<LINE  NUMBER>                           *AB          : ABORT
    A          : ASSEMBLE                    **B           : BREAK
    BC,BC1,BC2,BC3,BC4,BC5 : BREAK_CONDITION1,2,3,4,5
  **BS         : BREAK_SEQUENCE                CH          : CHECK
  **CL         : CLOCK                        *CC          : COMMAND_CHAIN
   *CV         : CONVERT                       DC          : DATA_CHANGE
    DS         : DATA_SEARCH                  *DA          : DISASSEMBLE
                                              *D           : DUMP
   *E          : END                           EM          : EXECUTION_MODE
    F          : FILL                          G           : GO
   *HE         : HELP                         *HT          : HISTORY
   *ID         : ID                            MP          : MAP
   *M          : MEMORY                        MD          : MODE
    MV         : MOVE                          MR          : MOVE_TO_RAM
                                              *P           : PRINT
    Q          : QUIT                         *RX          : RADIX
    R          : REGISTER                      RS          : RESET
    RT         : RESULT                       *SS          : SHORT_SYMBOL
                                              *ST          : STATUS
    S          : STEP                          SI          : STEP_INFORMATION
    SO         : STEP_OVER                    *SY          : SYMBOL
   *T          : TRACE                        *TC          : TRACE_CONDITION
   *TM         : TRACE_MEMORY                  TMO         : TRACE_MODE
   *TS         : TRACE_SEARCH                 *FCO         : FILE_COPY
   *FDI        : FILE_DIRECTORY               *FDU         : FILE_DUMP
   *FER        : FILE_ERASE                    FL          : FILE_LOAD
   *FRE        : FILE_RENAME                   FS          : FILE_SAVE
   *FTY        : FILE_TYPE                     FV          : FILE_VERIFY
   *FCH        : FLOPPY_CHECK                 *FF          : FLOPPY_FORMAT
    H          : HOST                          L           : LOAD
    SV         : SAVE                          TL          : TERMINAL
    TR         : TRANSFER                      V           : VERIFY
    IL         : INTFC_LOAD                    IS          : INTFC_SAVE
    IT         : INTFC_TRANSFER                IV          : INTFC_VERIFY
   *LAN        : LAN                           LH          : LAN_HOST
   *LO         : LOGOUT                       *FTP         : FTP
  *#OPEN       : OPEN                        *#LS          : LS
  *#PWD        : PWD                         *#ASC         : ASC
  *#BIN        : BIN                         *#STA         : STA
  *#CD         : CD                          *#BYE         : BYE
  *#CLOSE      : CLOSE                        #LL          : LAN_LOAD
   #LSV        : LAN_SAVE                     #LTR         : LAN_TRANSFER
   #LV         : LAN_VERIFY
```

Note: *: Usable in parallel mode
No *: Unusable in parallel mode
**: Available only for display in parallel mode
#: Available when the FTP server is open.

The above example is a display when E7000 is used. The display when using E7000PC is a little different.

2. To display each command format:

```
:HELP G  (RET)
```

Displays GO command format.

```
:
```

7.2.21	HISTORY HT	**Displays input command history**

Command Format

- Display : HISTORY (RET) (Displays all input commands)
  : HISTORY <history number> (RET) (Displays the input command of 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 input wait state. When the (RET) key is pressed, the displayed command is executed.

Example

```
:HISTORY (RET)
 1 MAP
 2 MAP  0 FFFFFF;U
 3 F 0 1000 FF
 4 B 300
 5 BC1 A=104
 6 HISTORY
:HISTORY 5 (RET)
:BC1 A=104  ------------Enters command input wait state
```

Restriction

Subcommands cannot be displayed by the HISTORY command.

7.2.22	ID ID	Displays emulator system program version

Command Format

- Display : ID (RET)

Description

- Display

  Displays the SH-2 emulator system version and revision numbers.

Example

To display the SH-2 emulator system version and revision numbers:

```
:ID (RET)
 SH7604 E7000 (HS7604EPD70SF) Vn.m.
 Copyright (C) Hitachi,LTD. 1993
 Licensed Material of Hitachi, Ltd.
:
```

7.2.23	**MAP** **MP**	**Specifies, displays, or cancels memory attribute(s)**

Command Format

- Specification : MAPΔ<start address>Δ<end address>;<memory attribute> (RET)
- Display : MAP[Δ<start address>Δ<end address>] (RET)
- Cancellation : MAP[Δ]–; (W/G) (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>:
U: Memory in the user system (cancels emulation memory usage)
S: Emulation memory in the emulation pod
SW: Emulation memory with write protection in the emulation pod
SG: Emulation memory with access inhibition in the emulation pod
W: Read only memory (write-protected) access (one area)
G: Guarded memory area (access-inhibited) (one area)

Description

- Specification

— Allocates the standard emulation memory to areas CS0 to CS3 in 128-kbyte units. The emulation memory can be write-protected or access-inhibited by specifying SW or SG, respectively. The start address is rounded down to 0 or a multiple of H'20000, and the end address is rounded up to a multiple of H'20000, minus one.

```
: MAP  0  H'1FFFF;S  (RET)
```

When a write-protected area is written to or an access-inhibited area is accessed by the user program, the user program execution is aborted.

To move the memory to the user system, specify option U. After allocation, the size of the unused standard emulation memory is displayed.

```
REMAINS EMULATION MEMORY S=D'xxxxkB

              D'xxxxkB  (Standard emulation memory)
```

Note that when standard emulation memory is allocated to areas CS0 to CS3, user system memory which is the same space as the allocated area cannot be correctly accessed.

— One write-protected area and one guarded memory area can be allocated to areas CS0 to CS3 in 128-kbyte units independently of the emulation memory allocation.

Write protected: Program execution stops when the area is written to by the user program

Access prohibited: Program execution stops when the program accesses (read/write) the area

With an emulator command, the user can read from and write to the write-protected area.

- Display

— Displays the memory attributes of the area defined by <start address> and <end address>, in the following format:

```
: MAP <start address>  <end address>  (RET)
  xxxxxxxx–xxxxxxxx;y          xxxxxxxx–xxxxxxxx;y                         (a)
  INTERNAL I/O  = xxxxxxxx–xxxxxxxx                                         (b)
  GUARDED AREA= xxxxxxxx–xxxxxxxx          xxxxxxxx–xxxxxxxx
  WRITE PROTECT AREA= xxxxxxxx–xxxxxxxx          xxxxxxxx–xxxxxxxx   (c)
  REMAINS EMULATION MEMORY    S=D'xxxxkB                                    (d)
```

(a) Address range and memory attributes

Displays the addresses in both the cache and through areas to which standard emulation memory is allocated.

y: Standard emulation memory attribute

U: User system memory

S: Standard emulation memory in the emulator pod

(b) Internal I/O address range

(c) Guarded memory area and write-protected area address ranges (displayed only when they are specified)

(d) Unused emulation memory size in decimal

S=D'xxxxkB (Standard emulation memory)

— When no address is specified, the memory attributes of the whole memory area are displayed in the format shown above.

- Cancellation

  Cancels the write protection and access inhibition specifications.

  : MAP – ;G (RET)　　　　Cancels all access inhibition specifications.

  To cancel write protection or access inhibition of the standard emulation memory (SW or SG), specify option S or U again.

Notes

1. If there is not enough standard emulation memory to satisfy the specification, the attribute is specified only for the memory area available.

2. Standard emulation memory cannot be allocated to areas other than CS0 to CS3.

3. Note that the memory contents are actually modified when an attempt was made to write to a write-protected or access-inhibited area by the user program and user program execution is aborted.

4. Cache area and through area, which are specified as standard emulation memory and as write protected and access prohibited, are treated as the same area. Therefore, if an attribute is allocated to one area, the other area automatically takes on the same attribute. Attribute display is performed for only the cache area.

5. Standard emulation memory can only be accessed in 16-bit bus cycles.

6. A memory attribute cannot be allocated to a range which extends into the cache, reserved, or through areas.

7. The area assigned to emulation memory area cannot be used as user memory. For example, if area CS0 is assigned to emulation memory, CS0 cannot be used as user memory. Areas CS1 to CS3, however, can be used as user memory.

Examples

1. To allocate standard emulation memory to the address range from H'1000000 to H'101FFFF:

```
:MP 1000000 101FFFF;S (RET)
 REMAINS EMULATION MEMORY  S=D'0384kB
:
```

2. To make the address range from H'4000000 to H'5FFFFFF to write-protected:

```
:MP 4000000 5FFFFFF ;W (RET)
:
```

7.2.24	MEMORY M	Displays or modifies memory contents

Command Format

- Display, modification : MEMORYΔ<modification address>[Δ<data>][;[<option>] [ΔN]] (RET)

<modification address>: Address of memory area to be displayed or modified
<data>: Data to be written to the address
<option>: Length of display or modification unit
B: 1 byte
W: 2 bytes
L: 4 bytes
O: Odd address, 1 byte
E: Even address, 1 byte
Default: 1 byte
N: No verification

Description

- Display, modification

— If the <data> is omitted, the emulator displays memory contents at the specified address and enters input wait state of the modified data. The user can then enter data and modify memory contents; this process can then be repeated for the next address. If option N is not specified, the data to be modified is read and verified. Data in the internal I/O area is never verified. Memory contents is displayed and modified data is input in the following format.

```
: MEMORY <address>  (RET)
  xxxxxxxx    yyyyyyyy    ?  [<data>][;<option>]  (RET)
```

xxxxxxxx: Address of data to be modified
yyyyyyyy: Memory contents displayed in modification units.
<data>: Modification 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.

Table 7.15 MEMORY Command Options

Option	Description
B	1 byte
W	2 bytes
L	4 bytes
O	Odd address, 1 byte
E	Even address, 1 byte
^	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 wait state.

```
: MEMORY  H'FFF0  H'F8 (RET)
:
```

Notes

1. This command can be executed in parallel mode, but the emulator does not operate in realtime. For details, refer to section 1.3, Realtime Emulation.

2. The internal I/O area must not be accessed in the access-inhibited size: do not access 8-bit areas in 16-bit (W) or 32-bit (L) units. If this is done, an address error will occur during user program execution.

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 write the data H'10 to address H'FE00 without displaying the memory contents:

```
:M FE00 10 (RET)
:
```

7.2.25	**MODE** **MD**	**Specifies or displays SH-2 MCU operating mode**

Command Format

- Specification : MODE;C (RET)
- Display : MODE (RET)

Description

- Specification

— Interactively specifies SH-2 MCU operating mode as shown below.

```
: MODE;C  (RET)
  E7000 MODE (MD5-0)  xx  ?   (a)  (RET)
  MODE SET (C:CONFIGURATION/U:USER/M:MASTER-SLAVE) = y ?  (b)  (RET)
  CONFIGURATION WRITE OK (Y/N) ?  (c)  (RET)
```

(a) Operating mode: Input hexadecimal values to specify MD5 to MD0 bits.
Example: Input 1 to select operating mode 1.

(b) SH-2 operating mode setting

C: SH-2 operating mode is set to the mode specified in the configuration file.

U: SH-2 operating mode is set to the mode specified with the operating mode selection pins (MD0 to MD5) on the user system.

M: Master or slave setting is selected by mode selection pin MD5 and the other setting (MD0 to MD4) is specified with the configuration file.

(c) Configuration file write confirmation message

Y: The specified parameters are written to the configuration file.

N: The specified parameters are not written to the configuration file and the command execution is terminated.

— When the E7000 is used, a writable system disk must be set up before MODE command execution. After the MCU operating mode is recorded in the configuration file, the E7000 can be initiated as the specified MCU operating mode.

When the E7000PC is used, the MCU operating mode is written in the configuration file in the directory where the current system program is stored on the IBM PC.

The emulator terminates after the MCU operating mode is set, and must then be restarted.

For details on SH-2 operating modes, refer to section 3.1, Clock Operating Mode Setting.

- Display

Displays the SH-2 operating mode, operating mode selection pin (MD0–MD5) status, master/slave mode setting, and operating mode setting on the user system in the following format:

```
: MODE  (RET)
  yyyyy MODE = xx  (MD5-0=nn)                    (a)
  MODE SET = z                                   (b)
```

(a) Operating mode (xx), operating mode selection pin status on the user system (MD5-0=nn) (refer to table 7.16) and current master/slave setting (yyyyy). If a value other than those shown in the table is displayed as nn, the SH-2 microcomputer does not operate correctly. Check the user system. When the user system is not connected, 3F is displayed.
yyyyy: MASTER: Master mode
SLAVE: Slave mode

Table 7.16 Operating Mode Selection Pin Status and Display

Master-Slave Mode	CS0 Area Bus Width		Clock Mode			
MD5	**MD4**	**MD3**	**MD2**	**MD1**	**MD0**	**Display (nn)**
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

(b) Operating mode setting (z)

z: C: SH-2 operating mode is set to the mode specified in the configuration file.

U: SH-2 operating mode is set to the mode specified with the operating mode selection pins (MD0 to MD5) on the user system.

M: Master or slave setting is selected by mode selection pin MD5 and the other setting (MD0 to MD4) is specified with the configuration file.

Notes

1. When a mode from 4 to 6 is selected as the clock mode, the emulator requires the clock signal to be input from the user system for correct operation. If the user system is not connected, set the clock mode from 0 to 3 with this command before restarting the emulator system.

2. If the operating mode selection pins (MD0 to MD5) are set to 3F, the SH-2 operating mode is set to the mode specified in the configuration file even when U or M is specified as the operating mode setting method.

Examples

1. To specify the operating mode as mode 2 and set the SH-2 operating mode to the mode specified in the configuration file:

```
:MODE;C (RET)
 E7000 MODE (MD5-0)   3F ? 2 (RET)
 MODE SET (C:CONFIGURATION/U:USER/M:MASTER-SLAVE)=M ? C (RET)
 CONFIGURATION WRITE OK (Y/N) ? y (RET)
 START E7000
  S:START E7000
  R:RELOAD & START E7000
  B:BACKUP FD
  F:FORMAT FD
  L:SET LAN PARAMETER
  T:START DIAGNOSTIC TEST
    (S/R/B/F/L/T)  ? S (RET)              (Emulator is re-activated by S)
```

2. To display the SH-2 operating mode status:

```
:MODE (RET)
 MASTER MODE=0B  (MD5-0=3F)
 MODE SET=M
:
```

7.2.26	**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>, <end address>, and <number of bytes> to <destination address>. Transfer is usually performed from the <start address>. However, if <destination address> is set within the range from <start address> to <end address> or <number of bytes>, transfer is performed from the <end address> or <start address> + <number of bytes>.

  — 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 character)
zz 'z':	Read data (hexadecimal and ASCII character)

  If data is to be written to a range including areas CS0 to CS3 and the cache data forced read/write area, only a partial write is performed excluding these areas.

Example

To transfer data in the address range from H'101C to H'10FC to address H'1000:

```
:MV 101C 10FC 1000  (RET)
:
```

7.2.27	MOVE_TO_RAM MR	Moves contents of ROM to standard emulation memory

Command Format

- Movement : MOVE_TO_RAMΔ<start address>Δ<end address>
  [;<memory attribute>] (RET)

<start address>: Start address of the ROM area to be moved
<end address>: End address of the ROM area to be moved
<memory attribute>: Type of standard emulation memory to be allocated
S: Standard emulation memory installed in emulator pod
SW: Standard emulation memory with write protection installed in emulator pod
Default: Standard emulation memory in the emulator pod (S)

Description

- Movement

— Use this command to temporarily modify ROM contents in the user system and execute the modified program. Transfers ROM contents to the specified standard emulation memory area. Data transfer to standard emulation memory is performed in 128-kbyte units. After data transfer, the unused standard emulation memory area is displayed as follows:

REMAINS EMULATION MEMORY S=D'xxxxkB

S=D'xxxx (Standard emulation memory)

— Only an amount of data as large as the unused standard emulation memory will be transferred.

— Contents of areas other than CS0 to CS3 cannot be transferred.

— Refer to section 7.2.23, MAP, for details on write-protected area settings.

Example

To allocate standard emulation memory to the address range from H'0 to H'1FFFF in the user system ROM area and transfer ROM contents:

```
:MR 0 1FFFF;S  (RET)
 REMAINS EMULATION MEMORY        S=D'0384kB
:
```

7.2.28	PRINT P	**Assigns or cancels output device for command result display (specific to the E7000)**

Command Format

```
•  Assignment    :  PRINT  (RET)                    (Selects printer)
                 :  PRINTΔ<file name>  (RET)        (Selects file)
•  Cancellation  :  PRINT[Δ]–  (RET)

                 <file name>:   Name of file where display information is to be stored
```

Description

- Assignment

— Assigns either the printer or a file (FD) to output command results, such as command inputs, execution results, and error messages.

— If the file already exists, the following message is displayed:

```
OVERWRITE (Y/N) ?   (a)  (RET)
```

(a) Y: Overwrites existing file.
N: Aborts the command.

— The following data are not output to the printer or file:

- Program counter during GO command execution
- HELP messages for each command
- Addresses being loaded, saved, or verified

— This command is specific to the E7000. To assign a printer in the E7000PC, use the logging function to the IBM PC files. For details, refer to section 3.7.2, Debugging Functions, in Part II, E7000PC Guide.

- Cancellation

Cancels the currently assigned printer or file.

Notes

1. Do not eject the floppy disk when a file is assigned.

2. To change the output destination from a printer or file that was previously assigned with this command, first cancel the assignment and then reassign the output to the new destination.

Examples

1. To assign a printer:

```
:P  (RET)
:
```

2. To assign the existing file SAMPLE.LOG:

```
:P SAMPLE.LOG  (RET)
 OVERWRITE(Y/N)   ? Y  (RET)
:
```

7.2.29	**QUIT** **Q**	**Terminates emulator system program**

Command Format

- Termination : QUIT [Δ<file name>[;S]] (RET)

  <file name>: Name of the file to contain emulation information (table 7.17)
  S: Option to save symbol information

Description

- Termination

— Terminates the emulator system program, puts the emulator monitor in command input wait state, and displays:

```
: QUIT  (RET)
  START E7000
    S:  START E7000
    R:  RELOAD & START E7000
    B:  BACKUP FD
    F:  FORMAT FD
    L:  SET LAN PARAMETER
    T:  START DIAGNOSTIC TEST
        (S/R/B/F/L/T)  ? _
                         ↑
                         Cursor
```

— Stores emulation information (shown in table 7.17) in the specified file and terminates the emulator system program. The emulation information can be restored later if the emulator system program is initiated with WARM START. Specify the S option to store symbol information. Note, however, that symbol information may require too big an amount of memory to be saved in a file.

```
: QUIT   <file name>  (RET)
```

If the specified file already exists, the following message is displayed:

```
OVERWRITE (Y/N) ? (a)(RET)
```

(a) Y: Overwrites existing file.

N: Aborts the command, and enters emulator system program command input wait state.

Table 7.17 Emulation Information Saved with the QUIT Command

Item	Description
Software breakpoints	Information set by the BREAK and BREAK_SEQUENCE commands
Hardware break conditions	Information set by the BREAK_CONDITION1,2,3,4,5 commands
Trace condition	Information set by the TRACE_CONDITION, TRACE_MODE, and TRACE_MEMORY commands
Memory map	Information set by the MAP command
Emulation operating mode	Information set by the EXECUTION_MODE command
Configuration information	Configuration file contents
Symbol information	Registered symbol information (Only available when the S option is specified)

Note

In the E7000, a file to store emulation information as listed in table 7.17 is created on a floppy disk. In the E7000PC, on the other hand, the file is created in the current directory. To store emulation information in another directory, specify that directory name.

Example

To save emulation information in the file SAMPLE.INF and terminate the emulator system:

```
:QUIT SAMPLE.INF (RET)
 START E7000
  S: START E7000
  R: RELOAD & START E7000
  B: BACKUP FD
  F: FORMAT FD
  L: SET LAN PARAMETER
  T: START DIAGNOSTIC TEST
     (S/R/B/F/L/T) ?  _
```

7.2.30	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 emulator system program initiation)
D: Decimal
Q: Octal
B: Binary

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 activation. Numbers may be entered in any radix at any time, provided that each value is prefixed with the appropriate character.

Examples:

Table 7.18 Radix and Input Examples

Radix	Input Example
Binary	B'1010
Octal	Q'2370
Decimal	D'6904
Hexadecimal	H'AF10

- 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

Note

The specified radix applies to the input of numeric values only; it does not affect display. To specify line number symbols, enter in decimal.

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
:
```

7.2.31	**REGISTER** **R**	**Displays register contents**

Command Format

- Display : REGISTER (RET)

Description

- Display

  Displays all register contents.

Example

To display all register contents:

```
:R  (RET)
 PC=0000FF02  SR=000003F3:MQIIII--ST
 PR=00000000  GBR=00000000  VBR=00000000
 MACH=00000000  MACL=00000000
 R0-7   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
 R8-15  00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFFFFC
:
```

7.2.32	**RESET** **RS**	**Resets SH-2**

Command Format

- Reset : RESET (RET)

Description

- Reset

  Resets the SH-2. The SH-2 general-purpose and control register contents will be reset to the following values:

R0 to R14	: The value before reset	VBR	: H'00000000
R15 (SP)	: Power-on reset vector value	GBR	: The value before reset
PC	: Power-on reset vector value	MACH	: The value before reset
SR	: H'000000F0 (I0 to I3 bits are 1)	MACL	: The value before reset
PR	: The value before reset		

  The internal I/O register contents will also be reset.

Example

To reset the SH-2:

```
:RS  (RET)
 ** RESET IN BY E7000!
:
```

7.2.33	**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=00001000   SR=000000F0: - - I I I I - - - -                                    (a)
 -PR=00000000   GBR=00000000   VBR=00000000
 -MACH=00000000      MACL=00000000
 -R0–7     00000000  00000001  00000002  00000003  00000004  00000005  00000006  00000007
 -R8–15    00000008  00000009  0000000A  0000000B  0000000C  0000000D  0000000E  0FFFFFFC
RUN-TIME=D'0000H:00M:00S:000018US[:000NS]                                           (b)
+++:<cause of termination >                                                         (c)
[LINE NO=<line number symbol>+n]                                                    (d)
```

(a) Register contents at program termination

(b) Execution time in decimal. Execution time can be measured up to 305 hours (1 µs minimum) or 76 hours (250 ms minimum) according to the TIME option setting of the EXECUTION_MODE command. Execution time exceeding 305 or 76 hours is displayed as an asterisk (*).

(c) Cause of termination. For a list of these causes, refer to section 7.2.19, GO, and section 7.2.36, STEP.

(d) If line number symbols are registered, the stop address is displayed as <line number symbol> +.

Note

Displayed register contents show values at program termination, not the current values.

Example

To display execution results:

```
:RT  (RET)
 -PC=0000FF02  SR=000000F0:--IIII----
 -PR=00000000  GBR=00000000  VBR=00000000
 -MACH=00000000   MACL=00000000
 -R0-7  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
 -R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFFFFC
 RUN-TIME=D'0000H:00M:00S:002241US[:000NS]
 +++:BREAK POINT   0000FF00
:
```

7.2.34	SHORT_SYMBOL SS	Defines a short format for a symbol and displays current short formats

Command Format

- Definition : SHORT_SYMBOLΔ\n = <short-format symbol> (RET)
- Display : SHORT_SYMBOL (RET)

n:	Number from 1 to 5
<short-format symbol>:	Abbreviated form of symbol

Description

- Definition

— Enables input of a short-format symbol, by defining a short format for the upper symbol or all symbols in a nested symbol sequence. Symbols in a nested sequence are delimited by slashes (/), as follows:

!<name>/<name>/<name> ...	Normal symbols
&<name>/<line number> ...	Line number symbols

Example:

```
: SHORT_SYMBOL  \1=MAIN/SUB  (RET)
: BREAK   !\1/ABC  (RET)
```

The above specification has the same meaning as the following:

```
: BREAK   !MAIN/SUB/ABC  (RET)
```

To define a short format for a symbol, the symbol must have already been defined.

— This command cannot delete the short format of a symbol, but the short format will be deleted if the corresponding symbol is deleted by the SYMBOL command.

- Display

Displays all previously defined short formats for review; blanks are displayed for symbols with no short formats.

Note

Short formats cannot be specified for <short-format symbol> in this command.

Examples

1. To specify a short format for a nested sequence of symbols and set a software breakpoint:

```
:SS \2=msym/main/cmd0  (RET)
:SS \3=msym/sub/cmd7  (RET)
:B !\2,!\3   (RET)
:
```

2. To display all current short formats:

```
:SHORT_SYMBOL  (RET)
 \1 = msym/sub/cmb1
 \2 = msym/sub/cmd2
 \3 =
 \4 =
 \5 =
:
```

7.2.35	**STATUS** **ST**	**Displays emulator execution status**

Command Format

- Display : STATUS (RET)

Description

- Display

Displays emulator execution status in the following format:

```
MODE=(a)      RADIX=(b)      BREAK=(c)      B_CND=(d)     T_CND=(e)
HOST=(f)    SYM=(g)                  LINE_SYM=(h)           STEP_INFO=REG:(i) /A:(j)
CLOCK=(k)   PRINT=(l)                              BASE=(m)    EML_MEM=S:(n)
```

(a) MODE=xx: SH-2 operating mode specified with the MODE command

(b) RADIX=xxx: Default input number type

BIN:	Binary
OCT:	Octal
DEC:	Decimal
HEX:	Hexadecimal

(c) BREAK=D'xxx: Number of breakpoints (decimal)

(d) B_CND=xxxxx: BREAK_CONDITION1,2,3,4,5 command setting. If specified, the specified number is displayed; otherwise, a blank is displayed.

(e) T_CND=xx: TRACE_CONDITION command setting

ST:	Subroutine trace mode
R:	Range trace mode
S:	Trace stop condition
T:	Trigger mode
NO:	No condition is set.

(f) HOST=$x_1x_2x_3x_4x_5$: Host interface protocol

x_1: Baud rate (BPS: Bits per second)

1: 2400 BPS 2: 4800 BPS 3: 9600 BPS 4: 19200 BPS 5: 38400 BPS

x_2: Data length for one character

7: 7 bits 8: 8 bits

x_3: Parity

E: Even O: Odd N: None

x_4: Number of stop bits

1: 1 stop bit 2: 2 stop bits

x_5: Busy control method

X: X-ON/X-OFF control R: RTS/CTS control

(g) SYM=D'xxxxx: Number of defined symbols (decimal)

(h) LINE_SYM=D'xxxxx: Number of defined line number symbols (decimal)

(i) STEP_INFO=REG:x_1 x_2 : Register information provided by the STEP command

x_1 1: Control register (PC, SR, PR, GBR, VBR, MACH, and MACL) information is displayed.

Space: No control register (PC, SR, PR, GBR, VBR, MACH, and MACL) information is displayed.

x_2 2: General-purpose register (R0 to R15) information is displayed.

Space: No general-purpose register (R0 to R15) information is displayed.

(j) /A:xxxxxxxx-xxxxxxxx: Memory address displayed with the STEP command.

(k) CLOCK=xx: Clock signal type

EML: Emulator internal clock

USER: User system clock

(l) PRINT=<file name>: File to which data is displayed on the console being output . If data is being output to a printer, #PR is displayed instead of <file name>. If no file name is specified, nothing is displayed.

(m) BASE=xx: Expansion function display

(n) EML_MEM=S:D'xxxxkB: Indicates the remaining size in the standard emulation memory.

xxxx: Remaining size in standard emulation memory

Example

To display the emulator status:

```
:ST  (RET)
 MODE=2   RADIX=HEX      BREAK=D'001      B_CND=1 3  T_CND=NO
 HOST=38N1X    SYM=D'00254    LINE_SYM=D'00786     STEP_INFO=REG:12/A:
 CLOCK=EML      PRINT=X1               BASE=00        EML_MEM=S:D'0512kB
:
```

7.2.36	**STEP** **S**	**Performs single-step execution**

Command Format

- Single step : STEP [Δ<number of execution steps>[Δ<start address>]]
  [;[<stop PC>][Δ<display option>][ΔI]] (RET)

<number of execution steps>: Number of steps to be executed (H'1 to H'FFFFFFFF).
Default: If <stop PC> and <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 the single-step execution is terminated. Default is <number of execution steps>.

<display option>: Specification of instructions to be displayed
J: Displays instruction and register contents only when a branch instruction is executed
R: Displays instruction and register contents only within the opening routine
Default: Displays instructions and register contents for all executed instructions

I: Interrupt permission during STEP command execution

Description

- Single step

— Performs single-step execution beginning at <start address>. The type of emulation performed (described below) depends on the specified parameters and option.

In addition, register and memory contents, address and instruction mnemonic information, and termination cause can be displayed in the following format:

(a)
```
PC=00001000   SR=000000F0: - -I I I I- - - -
PR=00000000   GBR=00000000  VBR=00000000
MACH=00000000    MACL=00000000
R0-7    00000000   00000001   00000002   00000003   00000004   00000005   00000006   00000007
R8-15  00000008   00000009   0000000A  0000000B   0000000C   0000000D   0000000E   0FFFFFFC
```

(b) <address>:<instruction mnemonic>

(c) MEMORY
<memory contents>

(d) +++: <cause of termination>

(a) Register information

(b) Address and mnemonic of the instruction that was executed

(c) Memory contents display

(d) 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. The causes are listed in table 7.19.

Table 7.19 Causes of STEP Command Emulation Termination

Message	Description
BREAK KEY	The BREAK key or (CTRL) + C keys were pressed.
STEP NORMAL END	The specified number of steps were executed.
STOP ADDRESS	The instruction at a stop PC was executed.
BREAK CONDITION 1	A break condition specified with the BREAK_CONDITION1 command was satisfied.
BREAK CONDITION 3	A break condition specified with the BREAK_CONDITION3 command was satisfied.
BREAK CONDITION 4	A break condition specified with the BREAK_CONDITION4 command was satisfied.
BREAK CONDITION 5	A break condition specified with the BREAK_CONDITION5 command was satisfied.
BREAK CONDITION 1,3,4,5	Break conditions specified with the BREAK_CONDITION1,3,4,5 command were satisfied.
GUARDED AREA ACCESSED	A guarded memory area was accessed.
WRITE PROTECT	A write-protected area was written to.
ILLEGAL INSTRUCTION	A break instruction (H'0000) was executed.
RESET IN BY E7000	A problem occurred in the user system. The emulator had input the RES signal for forced termination.
DMA GUARDED OR WRITE PROTECT	A write-protected area was written to or a guarded memory area was accessed by DMA.

- 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)

- 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.

  : 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, BTS, BFS, BRAF, and BSRF

- 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 an 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 the instruction mnemonic at the break 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.

Notes

1. Single-step execution is achieved by using the hardware break function. Accordingly, conditions specified with the BREAK_CONDITION2 command are invalid when using the STEP command.

2. Software breakpoints specified by 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_CONDITION1,3,4 command are satisfied, STEP execution may terminate without executing a single instruction.

5. If an interrupt such as an address error occurs before STEP command execution, the addresses and mnemonics of the executed instructions cannot be displayed correctly.

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=000000F0:--IIII----
 PR=00000000   GBR=00000000     VBR=00000000
 MACH=00000000  MACL=00000000
 R0-7  00000001 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15 00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001000                  MOV      R0, R1
 +++:STEP NORMAL END
:(RET)
 PC=00001004    SR=000000F0:--IIII----
 PR=00000000    GBR=00000000     VBR=00000000
 MACH=00000000   MACL=00000000
 R0-7  00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15 00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001002                  MOV      #00, R0
 +++:STEP NORMAL END
 :
```

2. To perform single-step execution from address H'1060 to H'1070 with information displayed only for branch instructions:

```
:S  FFFF 1060 :1070 J (RET)
 PC=0000106A  SR=000000F0:--IIII----
 PR=00000000  GBR=00000000  VBR=00000000
 MACH=00000000  MACL=00000000
 R0-7  0000106A 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15 00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001064                  JMP       @R0
 00001066                  NOP
 PC=0000106E  SR=000000F1:--IIII---T
 PR=00000000  GBR=00000000  VBR=00000000
 MACH=00000000  MACL=00000000
 R0-7  00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15 00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 0000106A                  BT        00001070
 PC=00001072  SR=000000F0:--IIII----
 PR=00000000  GBR=00000000  VBR=00000000
 R0-7  00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15 00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 MACH=00000000  MACL=00000000
 00001070                  NOP
 +++:STOP ADDRESS
:
```

7.2.37	STEP_INFORMATION SI	Specifies or displays information during single-step execution

Command Format

- Specification : STEP_INFORMATION[Δ<register information>][ΔA=<start address>
  [(Δ<end address>/Δ@<number of bytes>)]] (RET)
- Display : STEP_INFORMATION (RET)

<register information>: Register to be displayed
1: Displays PC, SR, PR, GBR, VBR, MACH, and MACL
2: Displays R0 to R15
ALL: All register information is output (default specification at emulator initiation)
–: No information displayed
Default: ALL
<start address>: Start address of memory dump
<end address>: End address of memory dump. (Default is 16 bytes of memory beginning at the start address.)
<number of bytes>: The number of bytes of memory dump. (Default is 16 bytes.)

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 range of register information and memory contents which are to be displayed.

```
(a) PC=00001004   SR=000000F0: - -IIII- - - -
    PR=00000000   GBR=00000000   VBR=00000000
    MACH=00000000   MACL=00000000
(b) R0-7    00000000  00000001  00000002  00000003  00000004  00000005  00000006  00000007
    R8-15   00000008  00000009  0000000A  0000000B  0000000C  0000000D  0000000E  0FFFFFFC
(c) 00001002                      MOV           #00, R0
(d) MEMORY
    0000FF80  00  04  00  FF  F0  00  02  00  10  00  02  00  0F  00  00  00  "................"
(e) +++:STEP NORMAL END
```

(a) Control register information (PC, SR, PR, GBR, VBR, MACH, and MACL)

(b) General-purpose register information (R0 to R15)

(c) Address and assembler mnemonic of executed instruction (Displayed in symbols, if symbols are registered.)

(d) Memory contents display

(e) Cause of termination

- Display

  Displays STEP information according to the specified contents. However, the address and assembler instruction mnemonic of each executed instruction are not displayed.

Note

An address error may occur for internal I/O area display because display of memory contents is performed by accessing in byte units.

Examples

1. To display only the contents of control registers (PC, SR, PR, GBR, VBR, MACH, and MACL) during STEP and STEP_OVER command execution:

```
:SI  1 (RET)
:
```

2. To not display register information during STEP and STEP_OVER command execution:

```
:SI - (RET)
:
```

3. To display memory contents from address H'FB80 to H'FB87 during STEP and STEP_OVER command execution:

```
:SI  A=FB80  FB87  (RET)
:
```

4. To display contents according to the specified display information:

```
:SI  (RET)
 PC=00001004 SR=000000F0:--IIII----
 PR=00000000 GBR=00000000  VBR=00000000
 MACH=00000000  MACL=00000000
 R0-7  00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15 00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001002                MOV        #00, R0
 MEMORY
 0000FB80  00 04 00 FF F0 00 02 00                        "............."
:
```

7.2.38	**STEP_OVER** **SO**	**Performs single-step execution except for subroutines**

Command Format

- Execution : STEP_OVER [<start address>][;I] (RET)

  <start address>: Address of the start of single-step execution. Default is the current PC address.

  I: Interrupt permission during single-step execution

Description

- Execution

— Performs single-step execution of instructions, except for subroutines called by the BSR, JSR, or BSRF instructions beginning at <start address>. If BSR, JSR, or BSRF is executed, acts as if the subroutine called by the BSR, JSR, or BSRF is a single instruction. If an instruction other than BSR, JSR, or BSRF is executed, register contents and the executed instruction are shown after each instruction is executed as in the STEP command.

— If BSR, JSR, or BSRF is executed, the STEP_OVER command sets a PC break at two instructions following the BSR, JSR, or BSRF instruction and executes the user program.

— During STEP_OVER command execution, register contents can be displayed in the following format. The range of register information and memory contents are displayed according to the STEP_INFORMATION command specifications.

```
(a) PC=00001004      SR=000000F0:- -I I I I- - - -
    PR=00000000      GBR=00000000    VBR=00000000
    MACH=00000000    MACL=00000000
    R0-7     00000000    00000001  00000002  00000003  00000004   00000005   00000006   00000007
    R8-15    00000008    00000009  0000000A  0000000B  0000000C   0000000D   0000000E   0FFFFFFC
(b) <address>:<instruction mnemonic>
(c) MEMORY
      <memory contents>
(d) +++:<cause of termination>
```

(a) Register information

(b) Address and mnemonic of the instruction that was executed

(c) Memory contents display

(d) 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.

— The software breakpoints (BREAK and BREAK_SEQUENCE specifications) and hardware breaks (BREAK_CONDITION1,2,3,4,5 specifications) 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 mnemonic of the instruction calling the subroutine are displayed.

Table 7.20 Causes of STEP_OVER Command Emulation Termination

Message	Description
BREAK KEY	The BREAK key or (CTRL) + C keys were pressed.
ONE STEP END	Single-step execution was completed.
SUBROUTINE END	Subroutine execution was completed.
GUARDED AREA ACCESSED	A guarded memory area was accessed.
WRITE PROTECT	A write-protected area was written to.
ILLEGAL INSTRUCTION	A break instruction (H'0000) was executed.
RESET IN BY E7000	A problem occurred in the user system. The emulator had input the RES signal for forced termination.

Notes

1. Single-step execution stops at the instruction immediately following a delayed branch instruction. Therefore, two instruction mnemonics are displayed.

2. This command must be used only when the program execution returns from a subroutine called by a BSR, JSR, or BSRF instruction to an instruction immediately following the BSR, JSR, or BSRF instruction.

3. If an interrupt such as an address error occurs before STEP_OVER command execution, executed instruction addresses and mnemonics cannot be displayed correctly.

Example

To execute a 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=000000F0:--IIII----
 PR=00000000  GBR=00000000 VBR=00000000
 MACH=00000000   MACL=00000000
 R0-7   00000001 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15  00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001000                 MOV       R0, R1
 +++:ONE STEP END
:(RET)
 PC=00001004  SR=000000F0:--IIII----
 PR=00000000  GBR=00000000 VBR=00000000
 MACH=00000000   MACL=00000000
 R0-7   00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15  00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001002                 MOV       #00, R0
 +++:ONE STEP END
:(RET)
 PC=0000100A  SR=000000F0:--IIII----
 PR-0000100A  GBR=00000000 VBR=00000000
 MACH=00000000   MACL=00000000
 R0-7   00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15  00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 00001006                 BSR       00002020  (Subroutine is not displayed.)
 00001008                 NOP
 +++:SUBROUTINE END
:(RET)
 PC=0000100C  SR=000000F0:--IIII----
 PR=0000100A  GBR=00000000   VBR=00000000
 MACH=00000000   MACL=00000000
 R0-7   00000000 00000001 00000002 00000003 00000004 00000005 00000006 00000007
 R8-15  00000008 00000009 0000000A 0000000B 0000000C 0000000D 0000000E 0FFFFFFC
 0000100A                 NOP
 +++:ONE STEP END
:
```

7.2.39	**SYMBOL** **SY**	**Defines, displays, or deletes symbol**

Command Format

- Definition : SYMBOLΔ!<symbol>=<address> (RET)
- Display : SYMBOL [Δ!<symbol>] (RET)
- Deletion : SYMBOL[Δ] – (RET)

!<symbol>: Symbol to be defined
<address>: Address of the symbol to be defined
—: Deletes all symbol definitions

Description

- Definition

— Defines a symbol; the attribute of the defined symbol is label.

```
: SYMBOL !LAB = H'FF00  (RET)
```

— The number of symbol definitions depends on the length of the symbol. Approximately 50,000 symbols can be defined with eight characters.

- Display

— Displays a defined symbol.

— If no symbol is specified, all symbols are displayed. If a nested symbol is specified, all symbols below that symbol are displayed as well.

- Display format is as follows:

```
LEV   SYMBOL   ATTRIBUTE    ADDRESS    SIZE   ELEMENT
 x    xx......x  xx.........x  xxxxxxxx   xxxx    xxxx
(a)     (b)        (c)          (d)       (e)     (f)
```

(a) Level number: Displays nesting level (0 to F) of the symbol.
(Symbols with nesting levels upper than F cannot be defined.)

(b) Symbol: Displays the symbol’s name. A nested symbol is displayed as a symbol for each nesting level. Each symbol starts in the column that corresponds to its attribute, as shown in table 7.21.

Table 7.21 Symbol Attributes and Related Display Start Columns

Attribute	Start Column
UNIT	0
PROCEDURE	1
STRUCTURE	2 or 3*
VARIABLE	2 or 3*
ARRAY	2 or 3*
LABEL	2
POINTER	2 or 3*

Note: * If the symbol is a structure element, its name starts in column 3.

(c) Attribute: Symbol attribute, as shown in table 7.22.

Table 7.22 Symbol Attribute Display

Attribute	Contents
UNIT	Unit
PROCEDURE	Function
STRUCTURE	Structure
VARIABLE	Variable
ARRAY	Array
LABEL	Label
POINTER	Pointer

(d) Address: Address of a symbol. For a structure element name, the offset value of the element from the start address of the structure is displayed. In this case, a plus sign (+) is placed in front of the address value.

(e) Size: One of the following sizes, depending on the attribute:

Pointer:	Pointer size
Structure:	Total structure size
Variable:	Variable size
Array:	Size of one element
Others:	Blank

(f) Number of elements: The number of elements is displayed only when the attribute is an array. A blank is displayed for other attributes.

- Deletion

  Deletes all symbols. However, symbols cannot be deleted if they are specified as breakpoints with the following commands:

  BREAK
  BREAK_CONDITION1,2,3,4,5
  BREAK_SEQUENCE
  TRACE_CONDITION
  TRACE_MEMORY

Note

The variable names of a structure must not be the same as the tag names. In the example below, change the variable names or omit the identifiers.

```
Example:  struct xyz {             (Tag name)
                  int  a;
                  int  b;
                }  xyz  ;          (Variable)
```

Examples

1. To define the symbol SUBOO:

```
:SY !SUBOO=450  (RET)
:
```

2. To display all symbols:

```
:SY  (RET)
 LEV   SYMBOL                 ATTRIBUTE     ADDRESS     SIZE   ELEMENT
  0       MAIN                LABEL         00001000
  0       LOOP                LABEL         00001100
  0       SUBA                LABEL         00003000
  0       SH703POD            LABEL         00000330
:
```

7.2.40	TRACE T	Displays trace information

Command Format

- Display : TRACE[Δ[–]<start pointer>[:[–]<end pointer>]][;[BP][ΔB]] (RET)

<start pointer>: Start pointer of trace display. (Default is the PTR option of TRACE_MODE.)

<end pointer>: End pointer of trace display. (Default is the PTR option of TRACE_MODE.)

–: A 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 TRACE_CONDITION or BREAK_CONDITION1 command.)

BP: Bus-cycle pointers specified as pointer values. (Default is the instruction pointer.)

B: Trace information is displayed in bus-cycle units. (Default is instruction mnemonic information.)

Description

- Display

— Displays trace information acquired during user program execution. Either instruction mnemonic or bus-cycle display format can be selected.

a. The B option specifies display in bus-cycle units.

: TRACE; B (RET)

b. If B is omitted, only instruction mnemonic information is displayed.

: TRACE (RET)

— 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 condition is satisfied (See note). Trace information acquired before the delay condition is satisfied is displayed with a minus (–). To specify a bus-cycle pointer, the BP option must be selected. The default is the instruction pointer.

Note: A delay starts to be counted from where a delay condition specified by the BREAK_CONDITION1 or TRACE_CONDITION command is satisfied. When no delay condition has been specified or termination has been caused by another reason, the current trace information will decide when the delay condition is to be satisfied.

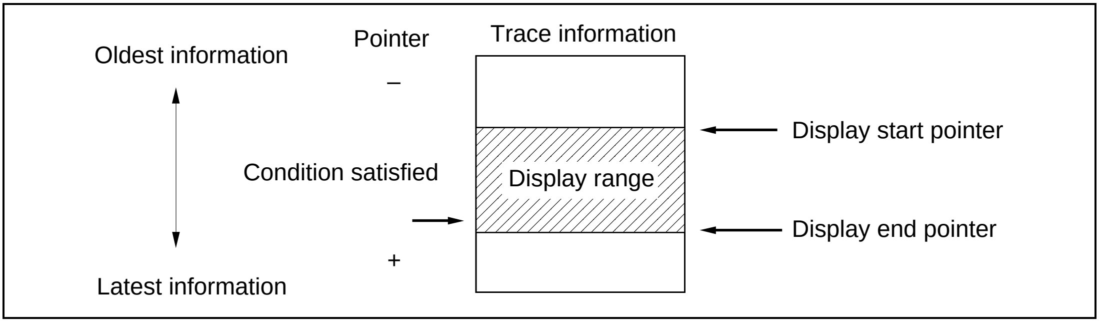


Figure 7.2 Display Range Specified by Instruction Pointers

Pointer default is as follows:

a. If <start pointer> is omitted, the start pointer specified by the PTR option of the TRACE_MODE command is used.

b. If <end pointer>is omitted, the end pointer specified by the PTR option of the TRACE_MODE command is used.

— To display only mnemonics of the executed instructions, uses the following format:

IP	ADDR	LABEL	MNEMONIC	OPERAND
* [–]D'xxxxx	00003010	!xx – xx	xx – xx	xx – xx
(a)	(b)	(c)	(d)	(e)

(a) Relative instruction location (instruction pointer), based on the instruction where a delay condition is satisfied as a break or trace condition. An instruction pointer begins with an asterisk (*) to differentiate it from a bus-cycle pointer.
Although the pointer usually has a negative value (–D'xxxxx), if a delay count condition is specified as a break or trace condition, the delay will be indicated as a positive value (D'xxxxx).

(b) Instruction address

(c) Label name

(d) Instruction mnemonic

(e) Instruction operand

— To display trace information in bus-cycle units, uses the following format:

```
    BP          AB        DB      MA  R/W  ST   IRL  NMI RES BRQ PRB VCC CLK TM
[-]D'xxxxx  xxxxxxxx  xxxxxxxx   xxx   x   xxx  xxxx  x   x   x   x   x   xx  xx
   (a)         (b)       (c)     (d)  (e)  (f)  (g)  (h) (i) (j) (k) (l) (m) (n)
-------------------------------------------------------------------------------
                                           TOTAL CLOCK NUMBER  =  xxxxx
                                                                   (o)
```

(a) Bus cycle pointer

Number of bus cycles since an instruction where a delay condition is satisfied. In bus cycles which prefetch instructions, the instruction mnemonics and instruction addresses are displayed as described above. When two instructions are executed in one bus cycle, both mnemonics are displayed, along with the address of the first instruction. Although the pointer usually has a negative value (-D'xxxxx), when a delay count condition is specified the delay will be indicated as a positive value (D'xxxxx).

(b) Address bus value

(c) Data bus value

Long-word, word, and byte values are displayed at the digits corresponding to the bus lines through which the data is accessed. For bus lines through which no data is accessed, an asterisk (*) is displayed.

(d) Memory area type

Table 7.23 MA Display

Display	Description
CAC	Cache hit
INT	Internal memory area access (H'40000000 to H'FFFFF800)
IO	Internal I/O area access
EXT	CS0 to CS3 access (including reserved area access)

(e) Read/write type

Table 7.24 R/W Display

Display	Description
R	Data read
W	Data write

(f) MCU status

Table 7.25 ST Display

Display	Description
PRG	Program fetch cycle (including PC relative data access cycle)
FIL	Cache fill cycle
DAT	Data access cycle (except for PC relative data access cycle)
DMA	DMA cycle
VCF	Vector fetch cycle

(g) IRL signal level for the SH-2

IRL0 to IRL3 signal level (0 = low level, 1 = high level)

```
   IRL
x3  x2  x1  x0
```

x3: IRL3 signal status	xn 0:	Low level
x2: IRL2 signal status	1:	High level
x1: IRL1 signal status		
x0: IRL0 signal status		

(h) NMI signal level (0 = low level, 1 = high level)

(i) RES signal level (0 = low level, 1 = high level)

(j) BREQ signal level (0 = low level, 1 = high level)

(k) External probe signal level (0 = low level, 1 = high level)

(l) Vcc voltage

Table 7.26 Vcc Voltage Display

Display	Description
0	Vcc voltage is less than 4 V; the SH-2 is not operating correctly
1	Vcc voltage is 4 V or more

(m) The number of clock cycles required from the end of the previous bus cycle to the end of this bus cycle. Up to 128 clocks are counted. If the number is more than 128, it is displayed as **. Refer to section 1.5.1, Trace Timing.

(n) Address specified by the TRACE_MEMORY command and its memory contents

(o) The total number of clock cycles (value displayed at (m) in the displayed trace range) displayed in hexadecimal. This value can be used to calculate the execution time in the displayed range. However, if ** (more than 128 clock cycles) appears at (m), ***** is displayed as the total value.

— If (CTRL) + P keys are pressed during trace information display, the emulator backs up 32 lines, displays 16 lines of data from that point, then stops display scrolling. At this point, if the (RET) key is pressed, the emulator resumes display scrolling. If (CTRL) + P keys are pressed again, the emulator will again back up 32 lines and display 16 lines of data. If the display is in bus-cycle units, the total number of clock cycles is taken from the display range specified at TRACE command input.

Notes

1. When the bus cycles are displayed, the following message is displayed as the emulator cycle following the last bus cycle of the user program execution. Note that this emulator cycle has no relation to the user program execution cycles.

   *** E7000 ***

   This cycle is being displayed even when a break occurs as a result of break conditions satisfied number of times, or as a result of a BREAK_SEQUENCE command.

2. When trace range is specified with the TRACE_CONDITION command, the executed assemble display is not correct. Accordingly, use the trace information display only for reference.

3. PC relative memory (MOV.W @(10,PC),R0 or so on) can be accessed in program fetch cycles (PRG is specified as the access type).

Examples

1. To display all trace information with only instruction mnemonics:

```
:T (RET)
    IP      ADDR        LABEL     MNEMONIC      OPERAND
 *-D'00004  00002010              JSR           @R0
 *-D'00003                        NOP
 *-D'00002  00002020              MOV.L         R0, @R1
 *-D'00001                        NOP
 * D'00000  00002024              MOV           R0, R4
:
```

2. To display trace information in bus-cycle units, from three instructions before the point where a break condition was satisfied:

```
:T -3;B  (RET)
      BP       AB     DB     MA  R/W  ST    IRL   NMI  RES  BRQ  PRB  VCC CLK TM
*          00002010                    JSR       @R0
*                                      NOP
-D'00005   00002010 400B0009 EXT  R   PRG  1111   1    1    1    1    1   03  FF
*          00002020                    MOV.L     R0, @R1
*                                      NOP
-D'00004   00002020 21020009 EXT  R   PRG  1111   1    1    1    1    1   03  FF
*          00002024                    MOV       R0, R4
-D'00003   00002024 6403000B EXT  R   PRG  1111   1    1    1    1    1   02  FF
-D'00002   0F000000 00002020 EXT  W   DAT  1111   1    1    1    1    1   01  00
-D'00001   00002028 00090009 EXT  R   PRG  1111   1    1    1    1    1   01  00
 D'00000     *** E7000 ***
---------------------------------------------------------------------------------
                                        TOTAL CLOCK NUMBER = 00000A
:
```

3. To specify a display range by bus cycle pointers, and display trace information in bus-cycle units:

```
:T -D'20:-D'16;BP B (RET)
     BP       AB        DB  MA  R/W  ST   IRL  NMI  RES  BRQ   PRB   VCC  CLK
*        00002014                    BRA       00002000
*                                    NOP
-D'00020 00002014 AFF40009  EXT  R   PRG  1111  1   1    1      1    1    02
*        00002000                    BRA       00002010
*                                    NOP
-D'00019 00002000 A0060009  EXT  R   PRG  1111  1   1    1      1    1    03
*        00002010                    JSR       @R0
*                                    NOP
-D'00018 00002010 400B0009  EXT  R   PRG  1111  1   1    1      1    1    03
*        00002020                    MOV.L     R0, @R1
*                                    NOP
-D'00017 00002020 21020009  EXT  R   PRG  1111  1   1    1      1    1    03
*        00002024                    MOV       R0, R4
*                                    RTS
-D'00016 00002040 6403000B  EXT  R   PRG  1111  1   1    1      1    1    02
-----------------------------------------------------------------------------
                                             TOTAL CLOCK NUMBER=00000D
:
```

TRACE_CONDITION

7.2.41	**TRACE_CONDITION** **TC**	**Specifies, displays, and cancels trace condition**

Command Format

- Setting : TRACE_CONDITION ΔA=<start address>:<end address>; ST (RET)
  (Subroutine trace)
  : TRACE_CONDITIONΔ<condition>[[Δ<condition>][Δ<condition>]....];R (RET)
  (Range trace)
  : TRACE_CONDITIONΔ<condition>[[Δ<condition>][Δ<condition>]....];T (RET)
  (Trigger trace)
  : TRACE_CONDITIONΔ<condition>[[Δ<condition>][Δ<condition>]....];S (RET)
  (Stop trace)
- Display : TRACE_CONDITION (RET)
- Cancellation : TRACE_CONDITION – (RET)

<start address>: Start address of subroutine trace
<end address>: End address of subroutine trace
<condition>: Range trace, trigger output, and trace stop conditions to be specified
ST: Subroutine trace mode condition
R: Range trace mode condition
T: Trigger mode condition
S: Trace stop condition

Description

- Setting

— Specifies trace acquisition condition (trace mode) for user program emulation (GO command execution).

Free trace: Acquires trace information during all bus cycles if no conditions have been set with this command.

Subroutine trace: Acquires trace information such as instructions and operands on the range (subroutine) specified by <start address> and <end address>. However, note that if the specified subroutine calls another subroutine, trace information on the called subroutine is not acquired.

Range trace: Acquires trace information during bus cycles in which the specified condition is satisfied.

Trigger trace: Acquires trace information during all bus cycles and a low-level pulse is output from the trigger pin of the emulator pod when 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.

— In subroutine trace, <start address> and <end address> must be specified following A=. In range trace mode, address or read/write condition can be specified as <condition>. In trigger or trace stop mode, the items shown in table 7.27 (other than external interrupt condition) can be specified as <condition> and they can be combined by ANDing them. Several conditions can be specified in any order.

Table 7.27 Specifiable Conditions (TRACE_CONDITION)

Item and Input Format	Description	Specifiable Trace Mode
Address condition	The condition is satisfied when the address bus value is in the range from <address 1> to <address 2>.	Range trace
• Range trace A=<address 1>[:<address 2>][;NOT] *	If <address 2> is omitted, the condition is satisfied when <address 1> is recognized. If NOT is specified, the condition is satisfied when an address other than the specified one is accessed. If an odd address was specified, access of an even address will satisfy the condition.	
• Trace stop or trigger trace A = <address>	The condition is satisfied when the address bus value matches <address>.	Trace stop or trigger trace

Data condition
D=<1-byte value>[:<bus width>]
WD=<2-byte value>[:<bus width>]
LD=<4-byte value>
<bus width> = 8, 16, or 32
	The condition is satisfied when the data bus value matches the specified value.
	D (byte access), WD (word access), or LD (long-word access) can be specified as a condition.
	For details, refer to the description in the following section.
	<bus width> specifies the bus width of memory to be accessed. Default is 16.
		Trace stop or trigger trace

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).	Range trace, trace stop, or trigger trace
Access type PRG: Program fetch cycle DAT: Execution cycle DMA: DMA cycle Default: All bus cycles described above	The condition is satisfied when the bus-cycle type matches the specified type. Multiple bus-cycle types cannot be specified. Make sure to specify one or none bus-cycle types.	Trace stop or trigger trace

Note: Refer to the description on the following page.

Table 7.27 Specifiable Conditions (TRACE_CONDITION) (cont)

Item and Input Format	Description	Specifiable Trace Mode
External probe condition PRB [:L] or PRB: H	The condition is satisfied when all the emulator’s external probe signal match the specified value. PRB or PRB: L: The condition is satisfied when the external probe signal is low PRB: H: The condition is satisfied when the external probe signal is high	Trace stop or trigger trace
System control signal condition BREQ	The condition is satisfied when the BREQ signal is low.	Trace stop or trigger trace
Delay count specification DELAY=<value> <value>: H'1 to H'7FFF	This condition can be specified in combination with any of the above conditions. The complete condition combination is satisfied when the specified number of bus cycles has been executed after the other specified condition is satisfied.	Trace stop or trigger trace
External interrupt condition NMI [:L] or NMI: H	This condition can be specified when the NMI signal matches the specified value. NMI or NMI: L: The condition is satisfied when NMI is low NMI: H: The condition is satisfied when NMI is high	Trace stop or trigger trace

Note: Refer to the description on the following page.

— Address and data conditions are satisfied when address bus values and data bus values match the specified values. Note the following when specifying trace conditions.

a Address and data conditions for trigger trace and trace stop modes is described below.

- Access to a 32-bit bus area (including internal ROM and RAM area)

  — Long-word access

  Long-word data is accessed in one bus cycle. Only long-word 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.
  Note that data conditions must be specified in combination with the address condition. If no address condition is specified or if the address is masked, data conditions will be satisfied when the address is a multiple of four.

  — 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. Note that data conditions must be specified in combination with the address condition. If no address condition is specified or if the address is masked, data conditions will be satisfied when the address is a multiple of four.

- Access to a 16-bit bus area

  — Long-word access

  Long-word data is accessed in two word-access cycles. Only long-word data (LD) and a multiple of two 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. Note that data conditions must be specified in combination with address condition. If no address condition is specified or if the address is masked, data conditions will be satisfied when the address is a multiple of two.

- Access to an 8-bit bus area

  All addresses can be accessed in byte units. Long-word data and word data are accessed in four byte-access cycles and two byte-access cycles, respectively. Both even and odd addresses can be specified as the address condition. Note, however, that only byte data (D) is valid for the data condition.

— A bit mask can be specified for address, data, or external probe conditions in trigger trace or trace stop mode. If a bit is masked, its value is always ignored and does not affect the condition satisfaction. To implement the mask, specify each digit to be masked at input as *. The number of masked bits depends on the input radix. Examples of masks are given below. Table 7.28 shows mask specifications.

Example: A condition is satisfied when the D0 bit is 0 in a byte data condition.

```
: TRACE_CONDITION  A = 3000000 D = B'*******0 (RET)
```

Table 7.28   Mask Specifications (TRACE_CONDITION)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	D0 and D5 bits are masked	Address, data (D, WD, LD), IRL
Hexadecimal	4 bits	H'F**50	D15 to D8 bits are masked	Address, data (D, WD, LD), IRL

Note: If <address 2> is not specified for a range trace, the lower bit of <address 1> can be masked but it is not possible to mask any desired bit position, as shown in the following examples.

Example:

```
Allowed:      TRACE_CONDITION A = H'10** ;R
Not allowed:  TRACE_CONDITION A = H'1*00 ;R
              TRACE_CONDITION A = H'100* :10** ;R
```

— 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. In this case, conditions that have been satisfied are all cleared, and the conditions are rechecked from the beginning. Old 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:

  A condition is newly set with the TRACE_CONDITION command

  TRACE command

  TRACE_SEARCH command

  As soon as the above command is terminated, trace information acquisition starts.
  In this case, conditions that have been satisfied Pare all cleared. Old information is also cleared. At this time,

  ```
  *** 81:TRACE CONDITION RESET
  ```

  is displayed.

- Display

Displays specified conditions. The specified input character string is displayed as is. If no condition is specified, blanks are displayed.

```
: TRACE_CONDITION (RET)
```

- Cancellation

Cancels specified conditions.

```
: TRACE_CONDITION – (RET)
```

Notes

1. When specifying an address condition, all address bits are checked. Therefore, if a trace condition is set at an address from the cache area, trace information will not be acquired when that address is accessed from the through area which occupies the same memory space. To acquire trace information at both the cache and through areas, mask the address specifications set with the TRACE_CONDITION command.

2. The bus-cycle condition is satisfied when PC relative memory (MOV.W @(10, PC), R0 and so on) is accessed in program fetch cycles (PRG is specified as the access type). Note that this specification differs from the BREAK_CONDITION1,2,3,4,5 command.

Examples

1. To acquire trace information only during the execution cycle of a subroutine in addresses H'4320 to H'4456 (subroutine trace):

```
:TC  A=4320:4456 ;ST (RET)
:
```

2. To acquire trace information for bus cycles when data is written to addresses in the range from H'2002 to H'2003 (range trace):

```
:TC  A=2002 W ; R (RET)
:
```

3. To stop trace and enter parallel mode if the external probe is pulled low (trace stop):

```
:TC  PRB=B'*******0 ; S (RET)
 ** TRACE STOP **
 # (command input wait state in parallel mode)
```

4. To output a trigger signal when byte data H'80 is written to address H'FF00 (trigger):

```
:TC  A=FF00 D=80 W ; T (RET)
:
```

7.2.42	**TRACE_MEMORY** **TM**	**Specifies, displays, and cancels memory address to be traced**

Command Format

- Setting : TRACE_MEMORY <address>[ΔDMA][;<size>] (RET)
- Display : TRACE_MEMORY (RET)
- Cancellation : TRACE_MEMORY [Δ]– (RET)

<address>: Memory address to be traced into trace buffer
DMA: Trace acquisition of DMA-cycle data. Default is an instruction execution cycle.
<size>: Traced size
8: 8-bit bus
16: 16-bit bus
32: 32-bit bus
Default: 16-bit bus

Description

- Setting

— Acquires the contents of the specified memory address in bus-cycle units and displays during user program execution.

— The address contents are displayed in byte units.

— Use the TRACE or TRACE_SEARCH command to display or search for the trace information. Specify the TM option of the GO command to display during user program execution.

- Display

Displays the specified address, access type, and size as shown below.

```
: TRACE_MEMORY (RET)
  ADDRESS = xxxxxxxx yyy;z
```

xxxxxxxx: Specified address
yyy: Access type
DMA: DMA cycle
Space: Instruction execution cycle
z: Traced size

- Cancellation

  Cancels the specified memory address.

Notes

1. Trace acquisition and display are performed in realtime. Specifications can be changed in parallel mode; however, note the following:

   - Data is traced and displayed in an instruction execution cycle or in a DMA cycle.
   - Data updated within the emulator, such as the internal I/O timer counter, cannot be updated until after the data has been accessed by instruction execution or DMA.
   - When an address for trace acquisition is changed while in parallel mode, the data collected up until then is for the previous address; the data will reflect the new address once this new address is accessed.
   - In parallel mode, the TRACE_MEMORY command setting cannot be cancelled.

2. The trace memory is modified when the specified address matches the address bus value. In this case, the A0 bit of the address bus is ignored but the A1 bit is valid. Accordingly, if the A1 bit of the specified address does not match the A1 bit of the address bus, the trace memory will not be modified, as shown in the following example.

   Example: If address H'F000000 is long-word accessed in the program while address H'F000002 is specified as an address condition, the trace memory is not modified.

3. Accessing PC relative memory (MOV.W @(10,PC),R0 and so on) will not update trace memory contents.

Examples

1. To trace the memory contents in address H'FF0E and display the results during user program execution:

```
:TM  FF0E;16 (RET)
:G ;TM (RET)
 ** PC = 00001204 0000FF0E = 80
```

2. To display the specified address:

```
:TM (RET)
 ADDRESS = 0000FF0E DAT ;W
:
```

3. To cancel the specification:

```
:TM - (RET)
:
```

7.2.43	TRACE_MODE TMO	Specifies and displays trace information acquisition mode

Command Format

- Setting : TRACE_MODE [ΔPTR=[–]<start pointer>[:[–]<end pointer>]][;C] (RET)
- Display : TRACE_MODE (RET)

<start pointer>: Start pointer during trace information display and search (Default at emulator initiation is –D'4095)

<end pointer>: End pointer during trace information display and search (Default at emulator initiation is D'4095)

C: Saves the specified mode in a configuration file

Description

- Settings

— Specifies the default values of start and end bus cycle pointers which are used when the bus cycle pointer values are not specified in the TRACE or TRACE_SEARCH command. Trace information in the emulator is available for a 32-kbyte bus cycle. 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 cycle pointer value ranges from –65535 to 65535. When exceeding this range, start and end pointers are automatically specified as –65535 and 65535, respectively.

```
: TRACE_MODE PTR = –D'2048:D'2048 (RET)
```

— Writes the specifications in a configuration file on the system disk when the C option is specified. Hereafter, when the emulator is activated with the system disk, the saved specifications go into effect. The following message is output to confirm with the user whether to rewrite the existing configuration file contents on the IBMPC.

```
: CONFIGURATION WRITE OK (Y/N) ?       (a) (RET)
```

(a) Y: Writes in the configuration file.
N: Does not write in the configuration file. The existing specifications are valid.

TRACE_MODE

- Display

Displays the specified trace mode as shown below.

```
: TRACE_MODE (RET)
  PTR = –D'yyyyy:D'yyyyy
```

yyyyy: Default values of start and end bus cycle pointers while trace information is displayed or searched

Examples

1. To display trace information excluding that on refresh cycles, specify the default values of the pointers within the range from –D'10 to D'10, and save the specified contents in a configuration file:

```
:TMO PTR= -D'10:D'10;C (RET)
  CONFIGURATION WRITE OK (Y/N)? Y (RET)
:
```

2. To display the specified contents:

```
:TMO (RET)
  PTR = -D'00010:D'00010
:
```

7.2.44	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 are displayed.

–: Specified when searching for trace information acquired before the trace or break condition has been satisfied. However, this specification can be omitted if delay count condition is specified in TRACE_CONDITION or BREAK_CONDITION1 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 specifications of the TRACE_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 the number of instructions saved in the trace buffer are displayed.

```
: TRACE_SEARCH (RET)
  INSTRUCTION NUMBER=D'xxxxx  BUS-CYCLE NUMBER = D'yyyyy
```

xxxxx: Number of instructions (decimal)

yyyyy: Number of bus cycles (decimal)

— Items listed in table 7.29 can be specified for <condition>, and they can be combined by ANDing them.

Table 7.29 Specifiable Conditions (TRACE_SEARCH)

Item and Input Format	Description
Address condition A=<address1>[:<address2>] *	Searches for the address bus value in the range from <address 1> to <address 2>. If <address 2> is omitted, only <address 1> is searched for.

Data condition
D=<1-byte value>
WD=<2-byte value>
LD=<4-byte value>
	Searches for a bus cycle where the data bus value matches the specified value.
	D (byte access), WD (word access), or LD (long-word access) can be specified as a condition.
	For details, refer to the corresponding description in the following pages.

Read/write condition R: Read W: Write	Searches for a read cycle (R is specified) or a write cycle (W is specified). For details, refer to the corresponding description in the following pages.
Access type PRG: Program fetch cycle (including PC relative data access cycle) DAT: Execution cycle DMA: DMA cycle FIL: Cache fill cycle VCF: Vector fetch cycle Default: All bus cycles described above	Searches for the specified type of bus cycle. Multiple bus-cycle types cannot be specified. Make sure to specify one or none bus-cycle types.
External probe condition PRB [:L] or PRB: H	Searches for a bus cycle in which the emulator external probe signal matches the specified value. PRB or PRB: L: The condition is satisfied when the external probe signal is low PRB: H: The condition is satisfied when the external probe signal is high

Table 7.29 Specifiable Conditions (TRACE_SEARCH) (cont)

Item and Input Format	Description
Memory area type CAC: Cache hit INT: Internal memory area access IO: Internal I/O area access EXT: Areas CS0 to CS3	Searches for a bus cycle in which the specified area is accessed.
External interrupt condition NMI: $\left\{\begin{array}{c}\text{H}\\\text{L}\end{array}\right\}$	• Searches for a bus cycle in which NMI is at the specified level. — NMI or NMI:L Searches for a bus cycle in which NMI is low. — NMI:H Searches for a bus cycle in which NMI is high.
IRL=<value>	• Searches for a bus cycle in which all IRL signals are at the specified levels. Specify <value> as one byte of data (H'0–H'0F) as follows. Each bit corresponds to a signal (IRL0 to IRL3). 3 2 1 0 ← Bit position x x x x ← Specified value | | | | 3 2 1 0 ← IRL number x: 0 = Low level 1 = High level Example: To search for a bus cycle in which IRL1 and IRL3 are high and others are low, specify: IRL=H'A Conditions NMI and IRL can be specified at the same time.
RES	Searches for a bus cycle in which RES is low.
Data condition for the address specified with TRACE_MEMORY command TM= $\left\{\begin{array}{c}\text{<data value>}\\\text{CH}\end{array}\right\}$	Searches for a bus cycle in which data in the address specified with the TRACE_MEMORY matches the specified value. If the CH option is specified, searches for a bus cycle in which data value changes.

— If the L option is specified, displays only the last bus cycle information to be searched for.

— When an address or data condition is specified, the emulator searches for a bus cycle where address bus value and data bus value match the specified values, respectively. Note the following when specifying search conditions.

a. Access to a 32-bit bus area (including internal ROM and RAM area)

- Long-word access

  Long-word data is accessed in one bus cycle. Only long-word data (LD) and a multiple of four can be specified as 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 addresses can be specified as the address condition.

b. Access to a 16-bit bus area

- Long-word access

  Long-word data is accessed in two word-access cycles. Only long-word data (LD) and a multiple of two 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.

c. Access to an 8-bit bus area

All addresses can be accessed in byte units. Long-word data and word data are accessed in four byte-access cycles and two byte-access cycles, respectively. Both even and odd addresses can be specified as the address condition. Note, however, that only byte data (D) is valid for the data condition.

— A bit mask can be specified for data, IRL, or external probe condition specification in single-bit or 4-bit unit. When a bit is masked, its value is always ignored and does not affect the condition satisfaction. To implement the mask, specify each digit to be masked at input as an asterisk (*). The number of masked bits depends on the input radix. Examples of masks are given below. Table 7.30 shows the mask specifications.

Example 1: To search for a bus cycle where D0 bit is 0 in byte data condition.

```
: TRACE_SEARCH  A = 4000000   D = B'*******0 (RET)
```

Example 2: To search for a bus cycle where IRL2 is 0 in IRL condition (IRL pins other than IRL2 are ignored)

```
: TRACE_SEARCH  IRL = B'*****0** (RET)
```

Table 7.30 Mask Specifications (TRACE_SEARCH)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	D0 and D5 bits are masked	Address, data (WD, D, LD), IRL
Hexa-decimal	4 bits	H'F**50	D15 to D8 bits are masked	Address, data (WD, D, LD), IRL

Note: A mask cannot be specified for address conditions.

— The display contents are the same as the bus-cycle display of the TRACE command. Refer to section 7.2.40, TRACE, for details. However, instruction mnemonics and the total number of clock cycles are not displayed.

— If no trace information satisfies the specified condition,

```
*** 45: NOT FOUND
```

is displayed.

— If trace information has not been saved in the trace buffer,

```
*** 39: BUFFER EMPTY
```

is displayed.

Note

1. The bus-cycle condition is satisfied when PC relative memory (MOV.W @(10,PC),R0 and so on) is accessed in program fetch cycles (PRG is specified as the access type). Note that this specification differs from the BREAK_CONDITION1,2,3,4,5 command.

2. Data bus information cannot be searched for when the vector fetch cycle is specified as the access type condition.

Examples

1. To search for bus cycles when data is written to addresses in the range from H'FF000000 to H'F000050:

```
:TS A=F0000000:F0000050 W  (RET)
     BP        AB        DB       MA  R/W  ST     IRL    NMI  RES  BRQ  PRB  VCC  CLK
 -D'00063  0F000003  ******44  RAM  W   DAT   1111    1    1    1    1    1    01
 -D'00062  0F000022  ****3344  RAM  W   DAT   1111    1    1    1    1    1    01
 -D'00060  0F000040  11223344  RAM  W   DAT   1111    1    1    1    1    1    01
:
```

2. To search for the last bus cycle where IRL0 is low:

```
:TS IRQ = B'*******0 ;L  (RET)
     BP        AB        DB       MA  R/W  ST     IRL    NMI  RES  BRQ  PRB  VCC  CLK
 -D'00063  0F000003  ******44  RAM  W   DAT   1111    1    1    1    1    1    01
:
```

Section 8 Floppy Disk Utility

8.1 Overview

The E7000 has a built-in 3.5-inch floppy disk drive which allows the user to save and load user programs and save execution results that are also output to the screen. Floppy disk utility commands are listed in table 8.1.

These commands cannot be used in the E7000PC.

Table 8.1 Floppy Disk Utility Commands

Command	Function	Usable/Unusable in Parallel Mode
FILE_COPY	Copies and verifies file contents	Usable
FILE_DIRECTORY	Displays file directory information	Usable
FILE_DUMP	Displays or modifies file contents	Usable
FILE_ERASE	Deletes file	Usable
FILE_LOAD	Loads file contents into memory	Unusable
FILE_RENAME	Renames file	Usable
FILE_SAVE	Saves memory contents to file	Unusable
FILE_TYPE	Displays file contents	Usable
FILE_VERIFY	Verifies file contents against memory	Unusable
FLOPPY_CHECK	Displays floppy disk information	Usable
FLOPPY_FORMAT	Formats a floppy disk	Usable

8.2 Floppy Disk Format

Format specifications of floppy disk for the E7000 are given in table 8.2. Prepare an appropriate 2HD floppy disk, and format it with the E7000’s FLOPPY_FORMAT command before using it on the E7000.

Table 8.2 Floppy Disk Format

Item	Specification
Medium	3.5-inch, 2HD (double sided, high density, double track)
Memory capacity	1.2 Mbytes (approx.)
Recording format	IBM format
Number of tracks	80 × 2 = 160
Number of sectors/track	15 sectors/track
Sector size	512 bytes/sector

8.3 Files

8.3.1 File Names

The general file name format is as follows:

<drive name>:<file name>.<extension>

File name specifications for the E7000 are given in table 8.3.

Table 8.3 File Name Specifications

Item	Number of Characters	Usable Characters	Default Enabled/Disabled	Default
Drive name	1	A, B	Enabled	A
File name	1 to 8	A to Z 0 to 9 $ & # { } ~ % – @ ^ ‘ ! _’ ( )	Disabled *	—
Extension	1 to 3		Enabled	(Space)

Note: * Default is enabled for some commands.

Drive Name: Since the E7000 has only one floppy disk drive, a drive name is usually not required with the file name. Drive A is the default drive. If drive B is specified for a copy procedure, copying to another floppy disk is assumed and a disk exchange message is displayed. When any commands other than the copy command are used, the drive need not be specified.

File Name: Identifies the file.

Extension: Identifies the file's attributes. Default is a space.

Wild card characters * and ? can be specified to select several files at a time with the FILE_DIRECTORY and FILE_ERASE commands. The meanings of these wild cards and examples of their use are given in table 8.4.

Table 8.4 Wild Card Characters

Wild Card Character	Meaning	Examples		
		Specification	Specified Files	
?	Regarded as a single character.	A?.COM	A.COM	AX.COM
		?.C?M	A.COM	B.CXM
		A???.???	A123.COM	ABCD.SYS
		????????.???	All files	
*	Regarded as a character string An asterisk (*) can be added to a complete file name or a complete extension XYZ.COM No character can be specified after *.	A?.*	A.COM	AB.CX
		ABC*.S	ABCD.S	ABCDEF.S
		*.COM	A.COM	
		.	All files	

8.3.2 File Configuration

Cluster: A file consists of clusters. A cluster contains 512 bytes and is the same size as a sector.

Record: Execution result files output with the PRINT command and text files that can be transferred from the host system are divided into records. The format of a record is shown below. One record can extend over several clusters.

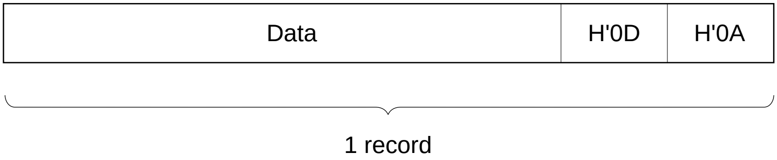
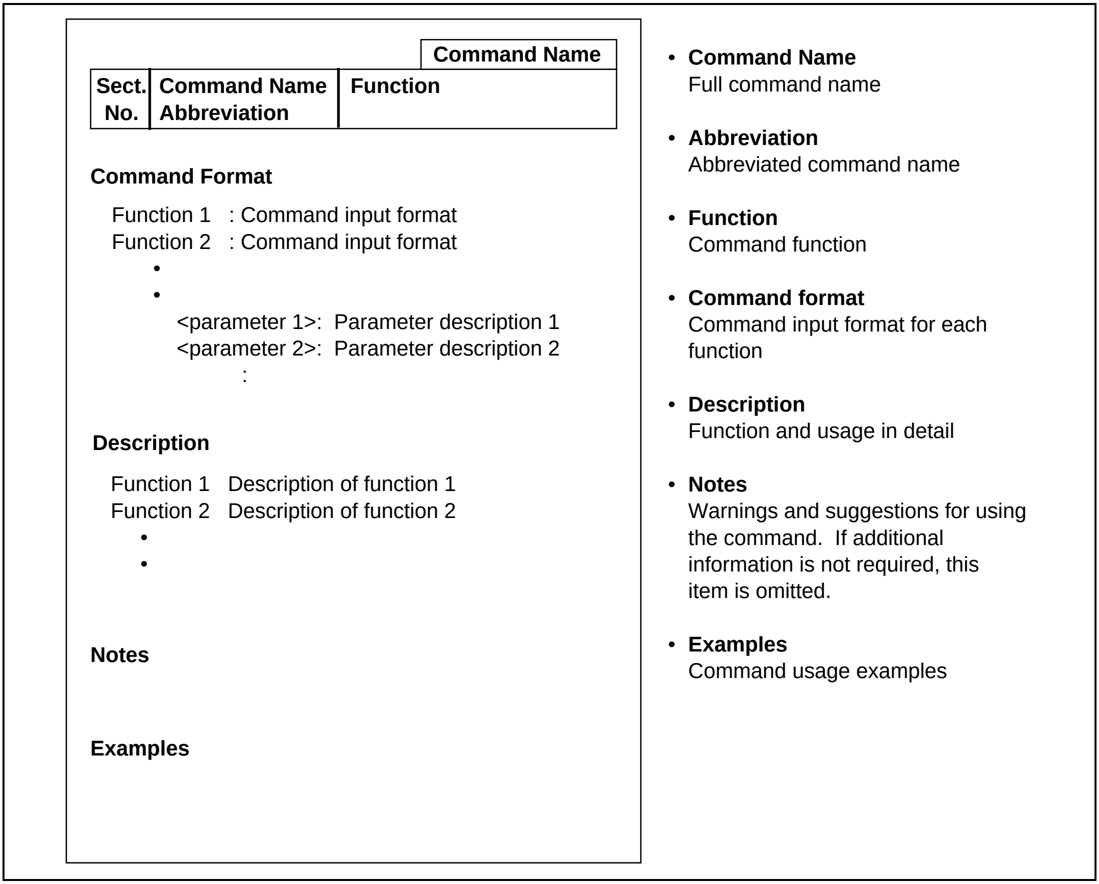


Figure 8.1 Record Format

Note: Only files that consist of records in the above format (with H'0D and H'0A at the end) can be transferred from the host system.

8.4 Floppy Disk Utility Commands

This section provides details of floppy disk utility commands in the format shown in figure 8.2.



		Command Name
Sect. No.	Command Name Abbreviation	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
- **Abbreviation**
  Abbreviated command name
- **Function**
  Command function
- **Command format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and suggestions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 8.2 Format of Floppy Disk Utility Command Description

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 are 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 underlining is used throughout this manual to indicate input, it is not used in the command format parts of these descriptions.

8.4.1	FILE_COPY FCO	Copies or verifies file contents

Command Format

- Copy :FILE_COPYΔ<source file>Δ<target file> (RET)
- Verification :FILE_COPYΔ<source file>Δ<target file>;(V/R) (RET)

V: Verifies files in byte units
R: Verifies files in record units

Description

- Copy

— Copies source file contents to the specified target file. When B: is specified as the target drive, the source file contents can be copied to another floppy disk. In this case, the disk exchange request message shown below is displayed after the source file's contents have been read from the floppy disk and stored in an internal memory buffer. Enter Y or N after exchanging the floppy disks.

```
SET TARGET FD OK (Y/N) ?   (a) (RET)
```

(a) Y: Copies contents of the source file from the internal buffer.
N: Aborts copy.

When copying to the same floppy disk, the drive name need not be specified.

— The short formats shown in table 8.5 can be used to specify the target file name.

Table 8.5 Target File Name Short Formats

Short Format	Specified File	Example
<file name>.<extension>	<file name>.<extension>	FCO F1.COM F2.COM
.<extension>	<source file name>.<extension>	FCO F1.COM .SA
B:	B:<source file name>.<source file extension>	FCO F1.COM B:
B:<file name>	B:<file name>.<source file extension>	FCO F1.COM B:F2
B:.<extension>	B:<source file name>.<extension>	FCO F1.COM B:.SA

— If the specified target file already exists, the response request message below is displayed. Enter Y or N.

```
OVERWRITE (Y/N) ?     (b) (RET)
```

(b) Y: Erases the existing file and creates a new one.
    N: Aborts copy.

- Verification

— Data is verified in byte units if option V is specified.

The verification error display format is as follows:

```
<CLUSTER>     OFFSET>      <SOURCE>     <TARGET>
   xxxx          xxx          xx 'x'       xx 'x'
   (a)           (b)           (c)          (d)
```

(a) Cluster number
(b) Offset from the beginning of the cluster
(c) Source file data in hexadecimal and ASCII characters
(d) Target file data in hexadecimal and ASCII characters

If the file sizes differ,

```
***  16:NOT SAME SIZE
```

is displayed.

— Data is verified in record units if option R is specified.

The verification error display format is as follows:

```
 <RECORD>     <OFFSET>     <SOURCE>     <TARGET>
   xxxx          xxx          xx 'x'       xx 'x'
   (a)           (b)           (c)          (d)
```

(a) Record number
(b) Offset from the beginning of the record
(c) Source file data in hexadecimal and ASCII characters
(d) Target file data in hexadecimal and ASCII characters

If the record sizes differ,

```
*** 16:NOT SAME SIZE
```

is displayed.

- When drive B is specified, the source file contents can be verified against target file contents on another floppy disk in the same way as for copy function. The verification must be specified in byte units (specify option V). Verification in record units is not possible.
- The short format of a target file name can be specified for the verify function in the same way as for the copy function (table 8.5).

Note

When the target file is on another floppy disk, the source file contents are first saved in the memory buffer. Therefore, the source file cannot be larger than the memory buffer’s capacity.

Examples

1. To copy file PROG.S to PROG.X on the same floppy disk:

```
:FCO PROG.S PROG.X (RET)
:
```

2. To copy file ABC.XYZ to another floppy disk by erasing a file of the same name and creating a new one:

```
:FCO ABC.XYZ B: (RET)
 SET TARGET FD OK (Y/N) ?Y (RET)
 OVERWRITE (Y/N)   ?Y (RET)
:
```

3. To verify files PROG.S and PROG.BAK in record units (two verification errors occurred):

```
:FCO PROG.S PROG.BAK;R (RET)
 <RECORD>   <OFFSET>   <SOURCE>  <TARGET>
 000031        00B       56 'V'    4C 'L'
 000031        00C       30 '0'    31 '1'
:
```

8.4.2	**FILE_DIRECTORY** **FDI**	**Displays file directory information**

Command Format

- Directory display :FILE_DIRECTORY[Δ<file name>] (RET)

<file name>: Specifies a file for directory information display. (Wild cards can be used.) If no file name is specified, all file directory information is displayed.

Description

- Directory display

— Displays directory information of the specified file. If <file name> is omitted, all file directory information is displayed.

The display format is as follows:

```
   <NAME>       <BYTES>     <NAME>       <BYTES>      <NAME>       <BYTES>
 xxxxxxxx.yyy    zzzzz    xxxxxxxx.yyy    zzzzz    xxxxxxxx.yyy    zzzzz
       :           :           :            :            :            :
VOLUME  LABEL  :  vvvvvvvvvv
```

xxxxxxxx: File name
yyy: File name extension
zzzzz: Number of bytes (in hexadecimal)
vvvvvvvvvv: Volume name

— The two wild card characters, ? and *, can be used in <file name>. For details on wild card characters, refer to section 8.3.1, File Names

?: Represents any single character or space.
*: Represents a character string.

Examples

1. To display all file directory information in a floppy disk:

```
:FDI (RET)
 <NAME>         <BYTES>    <NAME>      <BYTES>    <NAME>        <BYTES>
 E7000   .TST     F400     PROG   .S     4400      PROG   .BAK    4200
 TEST0001.S
 VOLUME LABEL : WORKFD
:
```

2. To display directory information of the files whose name starts with TST:

```
:FDI TST* (RET)
 <NAME>         <BYTES>    <NAME>       <BYTES>    <NAME>      <BYTES>
 TST001  .TST      100     TST002 .X       200     TSTAAA  .Y    4400
 TST0001 .S        120
 VOLUME LABEL : WORKFD
:
```

8.4.3	**FILE_DUMP** **FDU**	**Displays and modifies file contents**

Command Format

- Display :FILE_DUMP[Δ<file name>] (RET)
- Modification :FILE_DUMP[Δ<file name>]; I (RET)

<file name>: Specifies a file name. If the file name is omitted, the entire disk in drive A is assumed.

I: Subcommand input mode

Description

- Display

— Displays the contents of the specified file in cluster units (file dump). If the file name is omitted or drive A is specified, all data is displayed, beginning from the start sector of the floppy disk (floppy disk dump).

File dump:

```
DUMP OF <file name>     CLUSTER NO = <cluster number>
xxx   xx    xx    xx    ...........   xx    xx    xxxxxxxxxxxxxxxx
 .     .     .     .         .         .     .           .
 .     .     .     .         .         .     .           .
 .     .     .     .         .         .     .           .
xxx   xx    xx    xx    ...........   xx    xx    xxxxxxxxxxxxxxxx
 (a)        (b)                                         (c)
```

(a) Offset from that cluster

(b) Data in hexadecimal

(c) Data in ASCII characters

Floppy disk dump:

```
DUMP OF A:          SECTOR NO = <sector number>
xxx   xx    xx    xx    ...........    xx    xx    xxxxxxxxxxxxxxxx
 .     .     .     .         .          .     .           .
 .     .     .     .         .          .     .           .
 .     .     .     .         .          .     .           .
xxx   xx    xx    xx    ...........    xx    xx    xxxxxxxxxxxxxxxx
 (a)        (b)                                           (c)
```

(a) Offset from that sector

(b) Data in hexadecimal

(c) Data in ASCII characters

- Modification

When option I is specified, a prompt (>) is displayed and the E7000 enters subcommand input mode. Modify the file contents by inputting the subcommands shown in table 8.6. File contents are modified after reading the cluster or sector to be modified into the internal buffer with the R subcommand.

Table 8.6 FILE_DUMP Subcommands

Subcommand Format	Meaning
DΔ<start cluster> [<end cluster>] (RET)	Displays specified cluster data (or sector data)
RΔ<cluster number> (RET)	Loads the specified cluster data (or sector data) into the memory buffer
W (RET)	Writes memory buffer contents to the cluster (or sector)
M <start offset>Δ<end offset> (RET)	Displays specified memory buffer data
M <offset> (RET) xxx xx ? (a) (RET) (a) xx : Modification value ^ : Displays previous offset data . : Terminates modification mode (RET) only : Displays next offset data	Modifies memory buffer data. The specified offset address and its contents are displayed. Note: Cluster (or sector) data is not changed until the W command is input.
Q (RET)	Terminates the command

FILE_DUMP

Examples

1. To display the contents of file PROG.S:

```
:FDU PROG.S (RET)
 DUMP OF PROG.S              CLUSTER NO = 0000
 000   20 55 53 45 52 20 20 20  20 20 20 20 20 20 20 20  USER
 010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
 020   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
  :     :     :     :     :     :     :     :     :     :
```

2. To modify the contents of bytes 100 and 101 in sector 2 of file TEST.S:

```
:FDU TEST.S;I (RET)
>R 2 (RET)                  Loads sector 2 into memory buffer.
>M  100 (RET)               Displays the contents of byte 100.
>100 FF ? 80 (RET)          Changes the data in byte 100 from H'FF to H'80.
>101 00 ? 1 (RET)           Changes the data in byte 101 from H'00 to H'1.
>102 02 ? . (RET)           Terminates modification.
>W (RET)                    Writes memory buffer contents to the sector.
>Q (RET)                    Terminates FILE_DUMP command execution.
:
```

8.4.4	FILE_ERASE FER	Deletes file

Command Format

- Deletion :FILE_ERASEΔ<file name> [;Y] (RET)

```
<file name>:  File to be erased. (Wild card characters can be used.)
         Y:  Erases the file without a confirmation message
```

Description

- Deletion

— Deletes the specified file. If option Y is omitted, the confirmation message shown below is displayed. Enter Y or N.

```
  ERASE  <file name>            OK (Y/N)  ?  (a) (RET)
[ERASED <file name>]     (b)
```

(a) Y: Deletes the file.
N: Aborts command.
(b) This message is displayed when the file has been deleted.

If wild cards are used, all applicable files are processed as above.

— If option Y is specified, the file is deleted without the confirmation message. In that case, the following message is displayed:

```
ERASED  <file name>
```

— The two wild card characters, ? and *, can be used in <file name>. For details on wild cards, refer to section 8.3.1, File Names.

?: Represents any single character or space.
*: Represents a character string.

FILE_ERASE

Examples

1. To erase the file PROG.S (with a confirmation message):

```
:FER PROG.S (RET)
 ERASE   A:PROG    .S           OK (Y/N)  ?Y  (RET)
 ERASED  A:PROG    .S
:
```

2. To delete all files with extension LOG (without confirmation messages):

```
:FER *.LOG;Y (RET)
 ERASED  A:DB001   .LOG
 ERASED  A:TST     .LOG
 ERASED  A:PROG    .LOG
:
```

8.4.5	FILE_LOAD FL	Loads file contents into memory

Command Format

- Load :FILE_LOADΔ<file name>[Δ<offset>][;<load module type>] (RET)

<file name>: File to be loaded

<offset>: Value to be added to the address (can only be specified for S-type or HEX-type load modules) or start address (can only be specified for memory image file)
Default: H'0

<load module type>: Load module type to be loaded

S:	S-type load module
H:	HEX-type load module
B:	Binary-type load module
M:	Memory image file
Default:	Binary-type load module

Description

- Load

— Loads file contents saved by the FILE_SAVE command into user memory.

```
:FILE_LOAD <file name>[;<load module type>] (RET)
```

The current load address is displayed as follows. Note that the load address is not output to the file or to the printer assigned by the PRINT command.

```
LOADING ADDRESS  xxxxxxxx
```

xxxxxxxx: Current load address

When loading is completed, the start and end memory addresses are displayed as follows:

```
TOP ADDRESS = <start address>
END ADDRESS = <end address>
```

— If the load module is either S-type or HEX-type, an address offset (value to be added or subtracted) can be specified.

```
:FILE_LOAD <file name> <offset>;S (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

Notes

1. The program cannot be loaded into the areas other than CS0 to CS3 areas and cache data forced read/write area.

2. Verification is not performed after a file is loaded. The program must be verified with the FILE_VERIFY command, if necessary.

Example

To load the program saved in file SAVE.B by the FILE_SAVE command:

```
:FL  SAVE.B (RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00003FFF
:
```

8.4.6	FILE_RENAME FRE	Renames file

Command Format

- Renaming :FILE_RENAMEΔ<old file name>Δ<new file name> (RET)

Description

- Renaming

  — Replaces the old file name with the new file name. Short format can be used to specify the new file name. Table 8.7 lists short formats for specifying file names.

Table 8.7 Rename Short Formats

Short Format	Specified File	Example
<file name>.<extension>	<file name>.<extension>	FILE_RENAME F1.COM F2.CXT
.<extension>	<old file name>.<extension>	FILE_RENAME F1.COM .SXT
<file name>	<file name>.<old file extension>	FILE_RENAME F1.COM F2

Examples

1. To rename file PROG.SRC as TEST.S:

```
:FRE PROG.SRC TEST.S (RET)
:
```

2. To rename file PROG.SRC as PROG.SSS:

```
:FRE PROG.SRC .SSS (RET)
:
```

3. To rename file PROG.SRC as TEST.SRC:

```
:FRE PROG.SRC TEST (RET)
:
```

8.4.7	**FILE_SAVE** **FS**	**Saves memory contents to file**

Command Format

- Save :FILE_SAVEΔ<file name>Δ<start address>(Δ<end address>
  /Δ@<number of bytes>)[;<load module type>] (RET)

<file name>: File to be saved
<start address>: Start address of the memory area to be saved
<end address>: End address of the memory area to be saved
<number of bytes>: Number of bytes to be saved
<load module type>: Load module type to be saved
S: S-type load module
H: HEX-type load module
B: Binary-type load module
M: Memory image file
Default: Binary-type load module

Description

- Save

Transfers the contents of the specified memory area to the specified file, with the specified load module type. If the specified file already exists, the following response request message is displayed:

```
OVERWRITE (Y/N)  ? (a) (RET)
```

(a) Y: Overwrites the file.
N: Aborts the command without saving the file.

The current save address is displayed as follows. Note that the save address is not output to the file or the printer assigned by the PRINT command.

```
SAVING ADDRESS  xxxxxxxx
```

xxxxxxxx: Current save address

When the save is completed, the start and end memory addresses are displayed as follows:

```
TOP  ADDRESS = <start address>
END ADDRESS = <end address>
```

To reload the data which has been saved, use the FILE_LOAD command.

Notes

1. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be saved.

2. Verification is not performed after a file has been saved. The program must be verified with the FILE_VERIFY command, if necessary.

Examples

1. To save data from addresses H'0 to H'1FFF in the binary-type file SUB.B:

```
:FS SUB.B 0 1FFF (RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00001FFF
:
```

2. To save H'100 bytes of data from address H'4000 in the S-type file TST.S:

```
:FS TST.S 4000 @100;S (RET)
 TOP ADDRESS = 00004000
 END ADDRESS = 000040FF
:
```

8.4.8	FILE_TYPE FTY	Displays file contents

Command Format

- File contents display :FILE_TYPEΔ<file name> (RET)

Description

- File contents display

  Displays the contents of the specified file. The file contents must be written in valid ASCII characters. Since the FILE_TYPE command does not check the validity of ASCII characters, the screen will display unreadable text if invalid characters are included in the file. In addition, the file contents cannot be displayed if the records in the file do not have the format specified in section 8.3.2, File Configuration.

  The contents of execution result files output with the PRINT command can also be displayed with this command.

Example

To display the execution result file RESULT on the console:

```
:FTY RESULT (RET)
:MAP 0 1FFFF ;S
 REMAINS EMULATION MEMORY S=D'384kB
:F 0 FFFF
:FL TEST
 TOP ADDRESS = 000000
 END ADDRESS = 001FFF
:B 1020
:P -
:
```

8.4.9	FILE_VERIFY FV	Verifies file contents against memory

Command Format

- Verification :FILE_VERIFYΔ<file name>[Δ<offset>][;<load module type>] (RET)

<file name>:	Name of file to be verified
<offset>:	Value to be added to the address (can only be specified for S- or HEX-type load modules) or start address (can only be specified for memory image file) Default: H'0
<load module type>:	Load module type to be verified S: S-type load module H: HEX-type load module B: Binary-type load module M: Memory image file Default: Binary-type load module

Description

- Verification

— Verifies data in the specified file against that in user memory. The current verification address is displayed as follows:

```
VERIFY ADDRESS  xxxxxxxx

         xxxxxxxx:  Current verification address
```

Note that the verification address is not output to the file or to the printer assigned by the PRINT command.

When verification is completed, the start and end memory addresses are displayed as follows:

```
TOP  ADDRESS = <start address>
END ADDRESS = <end address>
```

If a verification error occurs, the following message is displayed:

```
<ADDR>      <FILE>      <MEM>
xxxxxxxx     xx  'x'     xx  'x'
  (a)         (b)         (c)
```

(a) Address where a verification error occurred

(b) File contents at the address where the error occurred (in hexadecimal and ASCII characters)

(c) Memory contents at the address where the error occurred (in hexadecimal and ASCII characters)

— If the load module is either S-type or HEX-type, an address offset (value to be added) can be specified.

```
:FILE_VERIFY <file name> <offset>;S (RET)
```

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

Note

Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be verified.

Examples

1. To verify data in file PRG.B against memory contents:

```
:FV PRG.B (RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00001FFF
:
```

2. To verify data against that in the S-type file TST.S (two verification errors occurred):

```
:FV TST.S (RET)
 <ADDR>    <FILE>   <MEM>
 00002024   30'0'    31'1'
 00003020   00'.'    01'.'
 TOP ADDRESS = 00002000
 END ADDRESS = 00003FFF
:
```

8.4.10	FLOPPY_CHECK FCH	Displays floppy disk information

Command Format

- Display :FLOPPY_CHECK (RET)

Description

- Display

  Displays the following information on the inserted floppy disk:

```
DRIVE A:
  xxx     BYTES/SECTOR                                (a)
  xxx     SECTORS/TRACK                               (b)
  xxx     TRACKS/DISK                                 (c)
TOTAL     AREA   xxxx SECTORS xxxxxx BYTES            (d)
FREE      AREA   xxxx SECTORS xxxxxx BYTES            (e)
```

(a) Number of bytes per sector (hexadecimal)

(b) Number of sectors per track (hexadecimal)

(c) Total number of tracks on disk (hexadecimal)

(d) Disk capacity in sectors and bytes (hexadecimal)

(e) Remaining capacity in sectors and bytes (hexadecimal)

Example

To display disk information:

```
:FCH (RET)
 DRIVE A:
   200 BYTES/SECTOR
   00F SECTORS/TRACK
   0A0 TRACKS/DISK
 TOTAL AREA 0960 SECTORS 12C000 BYTES
 FREE  AREA 06B3 SECTORS 0D6600 BYTES
:
```

8.4.11	**FLOPPY_FORMAT** **FF**	**Formats a floppy disk**

Command Format

- Format :FLOPPY_FORMAT (RET)

Description

- Format

Formats and initializes a floppy disk. Only use disks formatted with this command for the E7000. After the floppy disk is inserted and command is entered, the message below is displayed . Enter appropriate answers as follows:

```
:FLOPPY FORMAT (RET)
 VOLUME LABEL <volume label information> FORMAT OK (Y/N)  ? (a) (RET)
 *** FORMAT START ***      (b)
 KEY IN VOLUME LABEL (11 CHARACTERS)  ?(c) (RET)
```

(a) Specify whether or not the floppy disk is to be formatted.
Y: Formats the floppy disk.
N: Displays message. Moves to (c) without formatting the floppy disk.
When only initializing the floppy disk, enter N.
(b) Format start message.
(c) Enter a new label name (11 characters, max).

The floppy disk is initialized only when N is entered at (a). If the floppy disk is not to be formatted nor initialized, enter the (BREAK) key or the (CTRL) + C keys.

Note

Do not remove the floppy disk during this command execution. If a floppy disk is removed, an error occurs and the command execution terminates.

FLOPPY_FORMAT

Examples

1. To format and initialize a floppy disk. The volume name is E7000:

```
:FF (RET)
 VOLUME LABEL       FORMAT OK (Y/N)  ?Y (RET)
 *** FORMAT START ***
 KEY IN VOLUME LABEL (11 CHARACTERS)  ?E7000 (RET)
:
```

2. To initialize a floppy disk:

```
:FF (RET)
 VOLUME LABEL                  FORMAT OK (Y/N)  ?N (RET)
 KEY IN VOLUME LABEL (11 CHARACTERS)  ?WORK (RET)
:
```

Section 9 Data Transfer from Host System Connected by RS-232C Interface

9.1 Overview

When the E7000 is connected to a host system by the RS-232C interface, data can be transferred between the host system and the E7000 or between the host system and user system memory connected to the E7000. This enables the following transmission of host system load module files:

- Loads host system load module files to user system memory
- Saves user system memory data to host system files
- Transfers host system text files to E7000 floppy disk files
- Transfers E7000 text files to host system

Commands listed in table 9.1 can be used to transfer data.

These commands cannot be used in the E7000PC.

Table 9.1 E7000 Commands for Host System

Command	Function	Usable/Unusable in Parallel Mode
HOST	Specifies and/or displays host system interface parameters	Unusable
LOAD	Loads program from host system — Transparent mode and local mode	Unusable
SAVE	Saves program in host system — Transparent mode and local mode	Unusable
TERMINAL	Transfers to terminal mode — Transparent mode	Unusable
TRANSFER	Transfers file to and from host system — Transparent mode and local mode	Unusable
VERIFY	Verifies memory contents against host system file — Transparent mode and local mode	Unusable
INTFC_LOAD	Loads program from host system — Remote mode	Unusable
INTFC_SAVE	Saves program in host system — Remote mode	Unusable
INTFC_ TRANSFER	Transfers file to and from host system — Remote mode	Unusable
INTFC_ VERIFY	Verifies memory contents against host system file — Remote mode	Unusable

9.2 Host System Interface Modes and Operating Procedures

The E7000 interfaces with the host system in transparent mode, local mode, or remote mode, when RS-232C interface is used.

9.2.1 Transparent Mode

This mode is valid for a host system which is connected to the console by an RS-232C interface cable and is removable. A single console is shared between the E7000 and the host system in transparent mode.

The configuration and data transfer in this mode are shown in figure 9.1. To enable data transfer, host system commands must be entered in addition to emulator command.

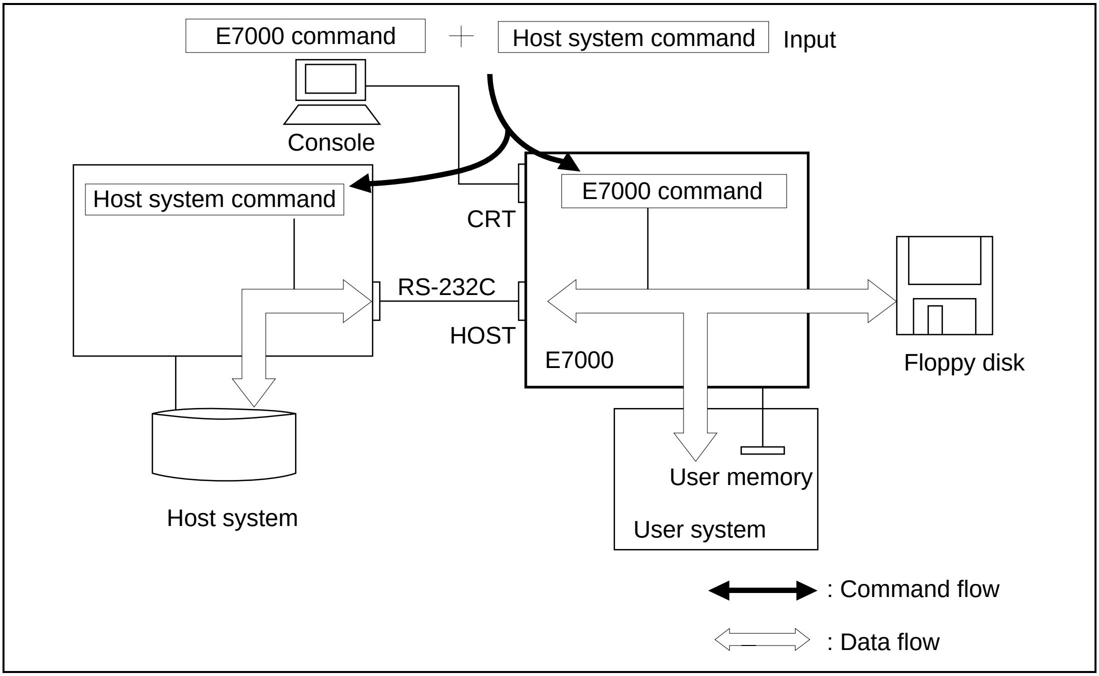


Figure 9.1 Configuration and Data Transfer in Transparent Mode

Procedure:

(a) Start up the E7000 system program

(b) Set up the host system’s communication parameters

Specify parameters such as baud rate and data length with the HOST command.

(c) Activate host terminal mode

Enter the TERMINAL command to make the console available to the host system.

(d) Set up host system

Start up the host system and put it in command input mode.

(e) Terminate host terminal mode

Exit host terminal mode with the code specified by the TERMINAL command.

(f) Input data transfer command

Add the host system command to the E7000 data transfer command (LOAD, SAVE, VERIFY, and TRANSFER)

Example: LOAD : TYPE X.MOT (RET)

9.2.2 Local Mode

In local mode, both E7000's console and host system's console are used for data transfer. An EPROM programmer can be connected in local mode. Emulator commands and host system commands are both executed at data transfer. The data receiving command must be executed first. The configuration and data transfer in this mode is shown in figure 9.2.

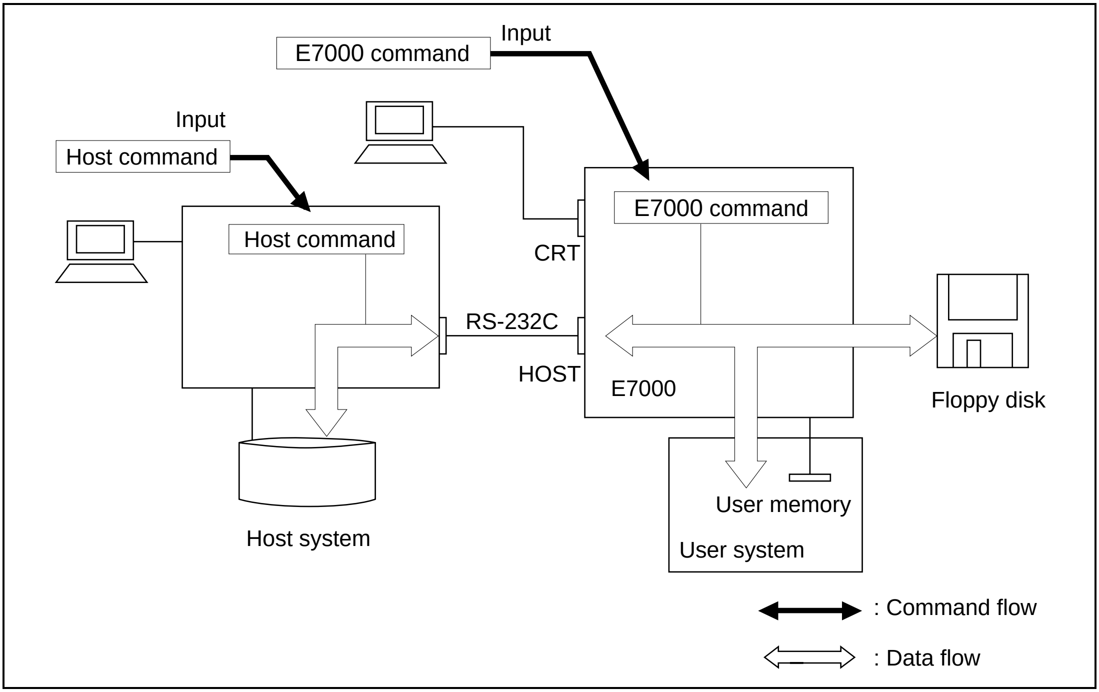


Figure 9.2 Configuration and Data Transfer in Local Mode

Procedure:

Host system	E7000	
Start up host system	Start up E7000 system program	
Set up communication parameters	Set up communication parameters	Specify parameters such as baud rate and data length with the HOST command.
<Data transfer from host system to E7000>		
	Execute data receive command	Execute the E7000 command (LOAD, VERIFY, or TRANSFER). The E7000 can receive data from the host system. Be sure to execute the receive command first.
Execute data transmission command		Execute the host system command to output data to the E7000 connection port.
<Data transfer from E7000 to host system>		
Execute data receive command		Execute the host system command to input data from the E7000 connection port.
	Execute data transmission command	Execute the E7000 command (SAVE or TRANSFER).

9.2.3 Remote Mode

In remote mode, a system that has its own console, such as a personal computer, is connected to the E7000 as the host system. This mode requires H-series interface software that can be separately purchased. In this mode, a host system's console can be used as the E7000's console. The data transfer commands in this mode are different from those in transparent mode and local mode: use the INTFC_LOAD, INTFC_SAVE, INTFC_VERIFY, and INTFC_TRANSFER in remote mode. The configuration and data transfer in this mode are shown in figure 9.3.

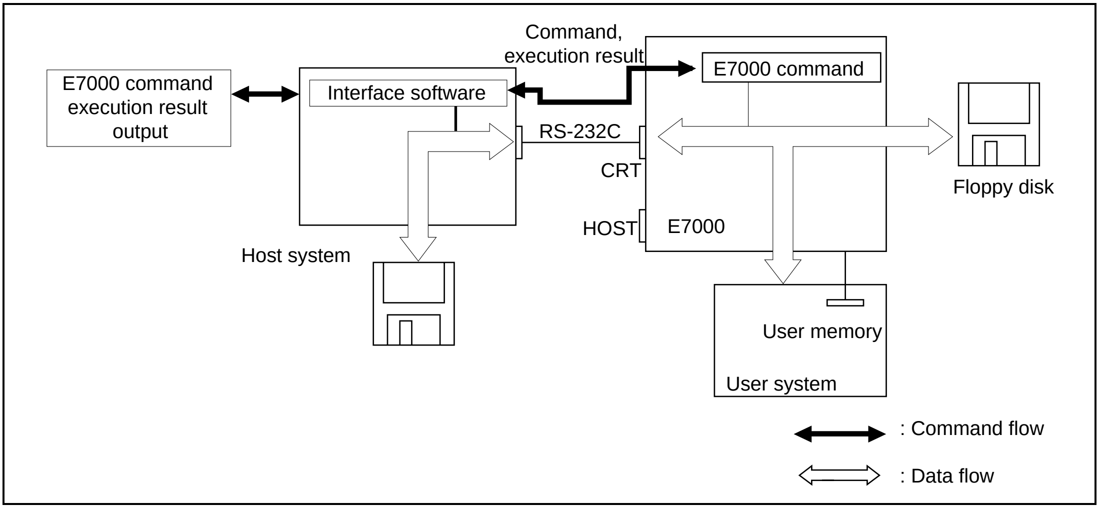


Figure 9.3 Configuration and Data Transfer in Remote Mode

Procedure:

Start up host system

Start up interface software

The following H-series interface software start-up message is displayed:

H-SERIES INTERFACE (type no.) Ver n.m
Copyright (C) Hitachi, Ltd. 1988
Licensed Material of Hitachi, Ltd.

Start up the E7000

The E7000 start-up message is displayed on the host system. E7000 commands can now be entered from the host system.

<Data transfer from host system to E7000>
Execute E7000 data receive command

The E7000 data receive command (INTFC_LOAD, INTFC_VERIFY, or INTFC_TRANSFER) can transfer data from the host system to the E7000.

Example:
INTFC_LOAD:<host system file name>

<Data transfer from E7000 to host system>
Execute E7000 data transmission command

The E7000 data transmission command (INTFC_SAVE or INTFC_TRANSFER) can transfer data from the E7000 to the host system.

Example:
INTFC_SAVE 0 1FFF:<host system file name>

9.3 Data Transfer Control

9.3.1 Control Methods

The E7000 provides an RS-232C interface for the host system. This interface supports the following two control methods that can be selected by the HOST command:

- X-ON/X-OFF control: Stops and restarts data transfer by the X-OFF (H'13) and X-ON (H'11) codes, respectively, sent from the data-receiving system.

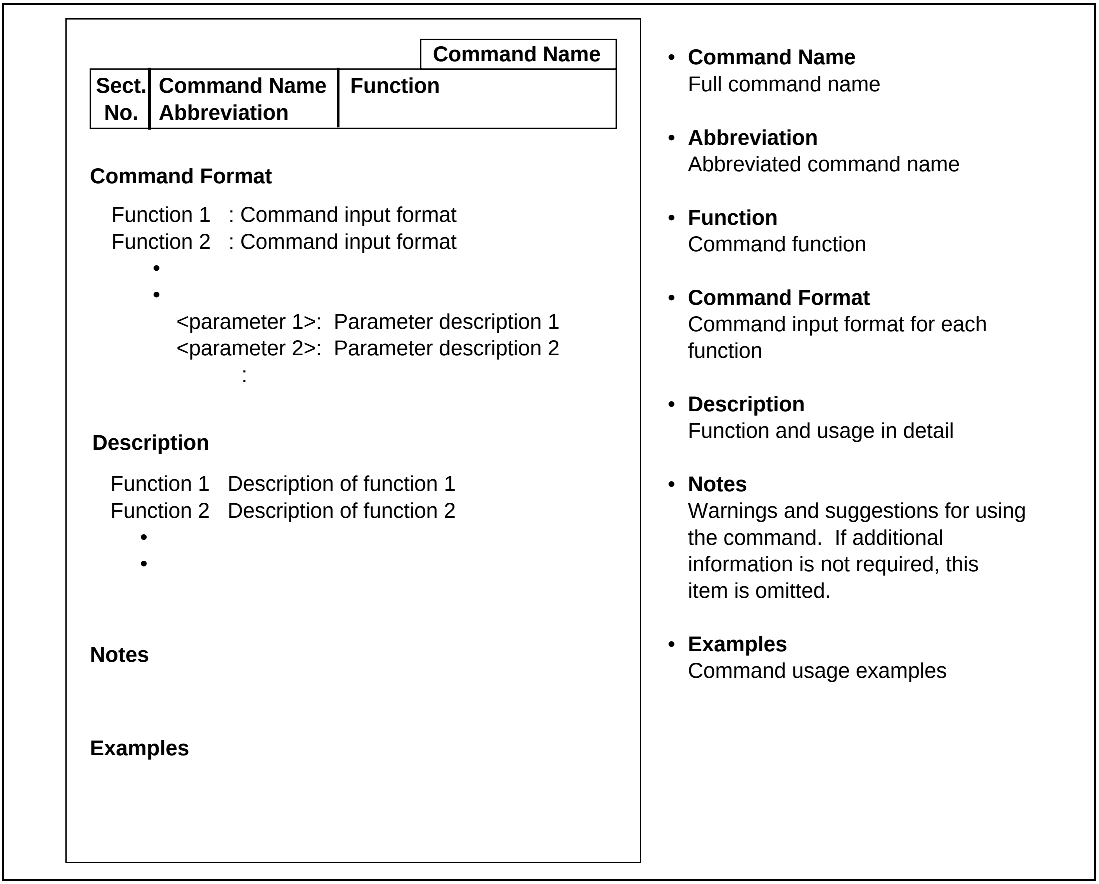
- RTS/CTS control: Stops data transfer when the data-receiving system outputs a low-level RTS signal, and restarts data transfer when the data-receiving system outputs a high-level RTS signal.

9.3.2 Timeouts

The E7000 monitors for timeouts as it receives data from the host system. After command execution, the E7000 waits an unlimited time for the first data byte. However, once it has received the first byte, it will timeout after waiting three seconds for the next data byte, and the command will terminate.

9.4 Host-System Related Commands

This section provides details of host-system related commands in the format shown in figure 9.4.



		Command Name
Sect. No.	Command Name Abbreviation	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
- **Abbreviation**
  Abbreviated command name
- **Function**
  Command function
- **Command Format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and suggestions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 9.4 Format of Host-System Related Command Description

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 are 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 underlining is used throughout this manual to indicate input, it is not used in the command format parts of these descriptions.

9.4.1	**HOST** **H**	**Specifies and/or displays host system communication parameters**

Command Format

- Specification :HOSTΔ<baud rate>Δ<character length>Δ<parity>Δ<stop bit>Δ <control method>(RET)
- Display and specification: HOST (RET)

<baud rate>:

1: 2400 bps
2: 4800 bps
3: 9600 bps
4: 19200 bps
5: 38400 bps

<character length>: Number of bits for one character

7: 7 bits
8: 8 bits

<parity>:

E: Even parity
O: Odd parity
N: No parity

<stop bit>: Number of stop bits

1: 1 stop bit
2: 2 stop bits

<control method>:

X: X-ON/X-OFF control
R: RTS/CTS control
For details, refer to section 9.3.1, Control Methods.

Description

- Specification

  Specifies the host system’s communication parameters (baud rate, character length, parity, stop bit, and control method). After this command is executed, data is transferred between the E7000 and the host system according to the interface conditions specified by this command.

- Display and specification

  Displays the current parameters and enables their respecification if necessary. Enter only (RET) to continue without changing the current parameter, a caret (^) to display the previous parameter again, and a period (.) to terminate the respecification.

```
:HOST (RET)
 BAUD RATE = 9600 BPS    CHARACTER LENGTH = 8    PARITY = NO PARITY   (a)
 STOP BIT = 2            CONTROL = X-ON,X-OFF

 BAUD RATE (1:2400/2:4800/3:9600/4:19200/5:38400) ? [n] (RET)          (b)
 CHARACTER LENGTH (7/8) ? [n] (RET)                                    (c)
 PARITY (E:EVEN/O:ODD/N:NO PARITY) ? [x] (RET)                         (d)
 STOP BIT (1/2) ? [n] (RET)                                            (e)
 CONTROL (X:X-ON, X-OFF/R:RTS, CTS) ? [x] (RET)                        (f)

 BAUD RATE = xxxxx BPS   CHARACTER LENGTH = x    PARITY = xxxxxxx      (g)
 STOP BIT = x            CONTROL = xxxxx
```

(a) Current communication parameters
(b) Baud rate
(c) Character length
(d) Parity
(e) Stop bit
(f) Control method
(g) Displays the selected communication parameters.

Note

At system initiation, the host system communication parameters are set according to the switches on the E7000’s control board. Refer to section 3.3, System Connection, in Part I, Emulator Guide, for details.

Examples

1. To specify all parameters at once:

```
:H 3 8 N 1 X (RET)
:
```

2. To specify parameters in interactive input mode:

```
:H (RET)
 BAUD RATE = 9600 BPS   CHARACTER LENGTH = 8   PARITY = NO PARITY
 STOP BIT = 1           CONTROL = X-ON,X-OFF
 BAUD RATE(1:2400/2:4800/3:9600/4:19200/5:38400) ? 4 (RET)
 CHARACTER LENGTH(7/8) ? 8 (RET)
 PARITY(E:EVEN/O:ODD/N:NO PARITY) ? E (RET)
 STOP BIT(1/2) ? 2 (RET)
 CONTROL (X:X-ON,X-OFF/R:RTS,CTS) ? X (RET)
 BAUD RATE = 19200 BPS   CHARACTER LENGTH = 8     PARITY = EVEN
 STOP BIT = 2            CONTROL = X-ON,X-OFF
:
```

9.4.2	LOAD L	Loads program from host system — Transparent mode and local mode

Command Format

- Load :LOAD[Δ<offset>][;[<load module type>][ΔN][ΔWA][ΔDEL]]
  [:<command transferred to host system>] (RET)

<offset>: Value to be added to the load module address (can only be specified for an S-type or HEX-type load module) or the start address (can only be specified for memory image file).
Default: H'0

<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
Default:	SYSROF-type load module

N: Specifies that <line number symbol> is not to be loaded. If omitted, <line number symbol> is loaded.

WA: Waits for the LF code. Refer to Description below.

DEL: Deletes all the symbols before loading. If omitted, does not delete any symbols.

<command transferred to host system>: Specifies a command to be transferred to the host system (only in transparent mode).

Description

- Load

— Loads a user program into user memory from the host system. Loading in different host system interface modes is described below.

Transparent Mode: After the command below is transferred to the host system, the user program from the host system is loaded into memory.

```
:LOAD;<load module type>:<command transferred to host system> (RET)
```

<command transferred to host system>: This command is transferred to the host system. The characters following the colon (:) are sent directly to the host system. The command to output source file contents to the terminal is specified.

The E7000 transfers <command transferred to host system> and (RET) (CR code: H'0D) to the host system. At the same time, it displays the echo back from the host system on the console. When option WA is not specified, the load module starts to transfer in 50 ms after (RET) is transferred. When option WA is specified, the E7000 waits for the LF code (H'0A) sent from the host system. As soon as the LF code is received, the load module starts to transfer. At first, try to transfer the load module without option WA. If it cannot be transferred, then specify option WA. If transfer does not occur with option WA, set the host system to no echo and transfer the load module without option WA.

Local Mode: The E7000 does not issue data output requests to the host system. Therefore, after the LOAD command is entered, set the host system to output data.

```
:LOAD[;<load module type>] (RET)
```

Remote Mode: Use the INTFC_LOAD command.

— The current load address is displayed in the format below.

```
LOADING ADDRESS = xxxxxxxx
```

xxxxxxxx: Current load address

When loading is completed, the start and end addresses are displayed as follows:

```
TOP ADDRESS = <start address>
END ADDRESS = <end address>
```

— If the load module is either S-type or HEX-type, an offset (value to be added) can be specified for the load module address.

```
:LOAD <offset>;S[:<command transferred to host system>] (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

— Information for symbolic debugging is included in a SYSROF-type load module. When a load module in SYSROF-type format is loaded, unit names of symbols to be defined can be selected as follows:

```
:LOAD;R[:<command transferred to host system>] (RET)
 ALL SYMBOL LOAD (Y/N)?  x (RET)  ............................. (a)
 LOAD UNIT NAME (name/.)  <unit name> (RET)  ............ (b)
    .         .         .         .              .
    .         .         .         .              .
    .         .         .         .              .
 LOAD UNIT NAME (name/.)  . (RET)  ............................... (b)
```

(a) Specifies whether all symbols are to be loaded or symbols are to be selected.

Y: Loads all symbols.

N: Enables the selection of symbols by unit name.

If Y is entered, all symbols are loaded and the confirmation request messages (b) are not displayed. If N is entered, the confirmation request messages are displayed.

(b) Symbol unit name to be defined

Loading starts when the period (.) is entered.

Up to ten unit names can be defined.

— If the N option is specified, <line number symbol> information among debugging information for the SYSROF-type load module is not loaded.

— Specifying the DEL option deletes all currently defined symbols.

```
:LOAD ; R DEL [:<command transferred to host system>] (RET)
```

However, if an attempt is made to delete all symbols including one currently in use (one used and specified with another command), the emulator displays the following message:

```
*** 30:SYMBOL IN USE
```

and terminates command execution without symbol deletion.

Notes

1. The load module cannot be loaded to the areas other than CS0 to CS3 areas and cache data forced read/write area.

2. Verification is not performed. The program must be verified with the VERIFY command if necessary.

3. The LOAD command reloads existing symbols to enhance throughput without checking for double definitions. When reloading the same load module, specify the DEL option or temporarily delete existing symbols before performing the LOAD command.

Examples

1. To load a SYSROF-type load module (transparent mode). COPYLINE F11.ABS TT: is a host system command. The symbol information for unit un001 will be loaded:

```
:L :COPYLINE F11.ABS TT: (RET)
 ALL SYMBOL LOAD (Y/N) ?  N (RET)
 LOAD UNIT NAME (name/.) ? un001 (RET)
 LOAD UNIT NAME (name/.) ? . (RET)
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

2. To load an S-type load module (local mode):

```
:L;S (RET)
                             ← After the LOAD command is entered, the host system
                               transfers data to the E7000.
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

9.4.3	SAVE SV	**Saves program in host system** **— Transparent mode and local mode**

Command Format

- Save :SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
  [;[<load module type>][ΔLF]][:<command transferred to host system>] (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 LF (H'0A) to the end of each record.
<command transferred to host system>: Specifies a command to be transferred to the host system (only in transparent mode).

Description

- Save

— Saves the specified memory contents in the host system in the specified load module type. An S-type or HEX-type load module can be saved. SYSROF-type and M-type load modules cannot be saved. Data receive request to the host system in different host system interface modes is described below.

Transparent Mode: After the specified command is transferred to the host system, the memory area contents of the specified load module type are saved in the host system.

```
:SAVE <start address> <end address>[;<load module type>]
                                   :<command transferred to host system> (RET)
```

<command transferred to host system>: This command is transferred to the host system. The characters following the colon (:) are sent directly to the host system. The command to save data sent from the terminal in a file is specified.

Local Mode: The E7000 does not issue data input requests to the host system. Therefore, before the SAVE command is input, set the host system to be ready to receive data.

```
:SAVE <start address> <end address>[;<load module type>] (RET)
```

Remote Mode: Use the INTFC_SAVE command.

— The current save address is displayed in the format below.

```
SAVING ADDRESS = xxxxxxxx
```

xxxxxxxx: Current save address

When save is completed, the start and end memory addresses are displayed as follows:

```
TOP ADDRESS=<start address>
END ADDRESS=<end address>
```

— When option LF is specified, the E7000 adds an LF code (H'0A) to the end of each record in addition to an CR code (H'0D) in the S- or HEX-type load module.

Notes

1. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be saved.
2. Verification is not performed. Verify the program with the VERIFY command if necessary.

Examples

1. To save memory contents in the address range from H'7000 to H'7FFF in the host system in S-type load module format (in transparent mode). COPY TT: F11.S is a host system command:

```
:SV 7000 7FFF :COPY TT: F11.S (RET)
 TOP ADDRESS=00007000
 END ADDRESS=00007FFF
:
```

2. To save memory contents in the address range from H'0000 to H'1E00 in the host system in HEX-type module format (in local mode):

← Before entering the SAVE command, set the host system to be ready to receive data from the E7000.

```
:SV 0 1E00 ;H (RET)
 TOP ADDRESS=00000000
 END ADDRESS=00001E00
:
```

9.4.4	TERMINAL TL	Transfers to terminal mode — Transparent mode

Command Format

- Transfer :TERMINAL [Δ<end code>] (RET)

  <end code>: End key code for TERMINAL mode (1-byte data).
  Default is H'1A ((CTRL)+ Z).

Description

- Transfer

  Transfers characters entered from the keyboard to the host system, and outputs data received from the host system to the console. A console connected to the E7000 can be used as a host system’s terminal, as shown in figure 9.5.

  Terminal mode ends when the specified end code is entered. Default is (CTRL) + Z (H'1A). This command is valid in transparent mode only.

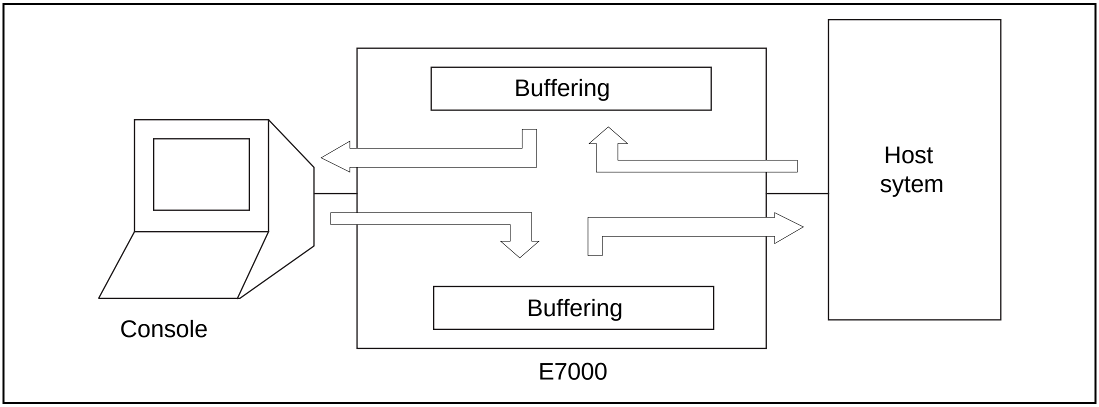


Figure 9.5 TERMINAL Command Processing

When changing the termination key code, specify hexadecimal data corresponding to the code, as follows:

(CTRL) + D: Specify H'4
(CTRL) + X: Specify H'18

Note

Terminal mode is controlled by software, although the terminal and the E7000 are connected by hardware. If the baud rate of the console interface is different from that of the host interface, the E7000 will still operate but some data may be lost. When the baud rate is 19200 bps or higher, (CTRL)+ S (display stop) may not be effective.

Example

To transfer to terminal mode with H'18 ((CTRL) + X) as terminal end code:

```
:TL 18 (RET)
$DIR            (Executes host system command)
     .             .               .                .
     .             .               .                .
     .             .               .                .
$(CTRL) + X     (Terminates terminal mode)
:
```

9.4.5	TRANSFER TR	**Transfers file to and from host system — Transparent mode and local mode**

Command Format

- Transfer :TRANSFER <file name>[;[(S/R)][∆WA][∆C]][:<command transferred to host system>] (RET)

<file name>: Name of file on floppy disk in the E7000
S: Transfer from the E7000 to the host system
R: Reception from the host system (default)
WA: Waits for the LF code after a command is transferred to the host system. Refer to Description below. Valid only when the R option is specified in transparent mode.
C: Inserts CR codes before LF codes in files transferred from the host system and removes CR codes from files transferred to the host system.
<command transferred to host system>: Specifies a command to be transferred to the host system (only in transparent mode).

Description

- Transmission

— Only text files can be transferred from the E7000 to the host system. Transfer processing in different host system interface modes is described below.

Transparent Mode: After the host system command below is transferred, file contents are sent to the host system.

```
:TRANSFER <file name>;S :<command transferred to host system> (RET)
```

<command transferred to host system>: This command is transferred to the host system. The characters following the colon (:) are sent directly to the host system. The command to store data from a terminal to a file is specified.

Local Mode: The E7000 does not issue data output request to the host system. Therefore, before this command is entered, set the host system to input data from the E7000.

```
:TRANSFER <file name>;S (RET)
```

Remote Mode: Use the INTFC_TRANSFER command.

- Receive

— The transfer of a file from the host system to the E7000. Only text files can be transferred. Data transfer processing in different host system interface modes is described below.

Transparent Mode: After the host system command below is entered, transferred data is written to the E7000 file.

:TRANSFER <file name>;R :<command transferred to host system> (RET)

<command transferred to host system>: This command is transferred to the host system. The characters following the colon (:) are sent directly to the host system. The command to output the file contents to a terminal is specified.

The E7000 transfers <command transferred to host system> and (RET) (CR code: H'0D) to the host system. At the same time, it displays the echo back from the host system on the console. When option WA is not specified, the load module starts to transfer in 50 ms after (RET) is received. When option WA is specified, the E7000 waits for the LF code (H'0A) sent from the host system. As soon as the LF code is received, the load module starts to transfer. At first, try to transfer the load module without option WA. If it cannot be transferred, then specify option WA. If transfer does not occur with option WA, set the host system to no echo and transfer the module without option WA.

Local Mode: The E7000 does not issue data output requests to the host system. Therefore, after the TRANSFER command is entered, set the host system to output data.

:TRANSFER <file name>;R (RET)

Remote Mode: Use the INTFC_TRANSFER command.

If the specified file already exists, the message below is displayed. Enter Y or N.

OVERWRITE (Y/N) ? (a) (RET)

(a) Y: Overwrites the existing file with the new file.
N: Aborts the command.

— The E7000 can receive only text files (ASCII characters) that can be displayed. If the E7000 receives another type of data, an error is generated and the command is aborted.

This command is terminated with EOF (H'1A).

— With a UNIX-based host system, each record is terminated with a single LF code and no CR code. To receive such records, specify option C.

Examples

1. To output a file from the E7000 to the host system in transparent mode. COPY TT: SAMPLE.S is a host system command:

```
:TR SAMPLE.S;S:COPY TT: SAMPLE.S (RET)
:
```

2. To transfer a file to the host system (in local mode):

← Before entering the TRANSFER command, set the host system to be ready to receive data from the E7000.

```
:TR FILE.TXT;S (RET)
:
```

3. To receive a file from the host system in transparent mode. TYPE FILE.S is a host system command:

```
:TR FILE.TXT;R:TYPE FILE.S (RET)
:
```

4. To receive a file from the host system (in local mode):

```
:TR FILE.TXT;R (RET)
```

← After entering the TRANSFER command, set the host system to output data to the E7000.

```
:
```

9.4.6	VERIFY V	Verifies memory contents against host system file — Transparent mode and local mode

Command Format

- Verification :VERIFY [Δ<offset>][;[<load module type>][ΔWA]][:<command transferred to host system>] (RET)

<offset>: Value to be added to the address (can only be specified for a S-type or HEX-type load module) or the start address (can only be specified for M-type load module). If the start address is omitted, 0 is assumed.

<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
Default:	SYSROF-type load module

WA: Waits for the LF code after a command is transferred to the host system. Refer to Description below.

<command transferred to host system>: Specifies a command to be transferred to the host system (only in transparent mode).

Description

- Verification

  — Verifies data transferred from the host system against data in memory. Verification in different host system interface modes is described below.

  Transparent Mode: After the host system command below is transferred, the user program from the host system is verified against the memory contents.

```
:VERIFY;<load module type>:<command transferred to host system> (RET)
```

<command transferred to host system>: This command is transferred to the host system. The characters following the colon (:) are sent directly to the host system. The command to output the file contents to the terminal must be specified.

The E7000 transfers <command transferred to host system> and (RET) (CR code: H'0D) to the host system. At the same time, it displays echo back from the host system on the console. When option WA is not specified, the emulator starts to receive a load module within 50 ms after (RET) is transferred. When option WA is specified, the E7000 waits for the LF code (H'0A) sent from the host system. As soon as the emulator receives the LF code, the emulator starts to receive the load module. At first, try to transfer the load module without option WA. If it cannot be transferred, then specify option WA. If transfer does not occur with option WA, set the host system to no echo and transfer the load module without option WA.

Local Mode: The E7000 does not issue data output requests to the host system. Therefore, after the VERIFY command is entered, set the host system to output data.

Remote Mode: Use the INTFC_VERIFY.

— 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)

— If the load module is either S-type or HEX-type, an address offset (value to be added or subtracted) of the load module can be specified.

```
:VERIFY <offset>; S [:<command transferred to host system>] (RET)
```

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

Notes

1. Symbolic data cannot be verified.

2. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be verified.

Examples

1. To verify a SYSROF-type load module against the memory contents in transparent mode. COPYLINE F1.ABS TT: is a host system command:

```
:V :COPYLINE F1.ABS TT: (RET)
 <ADDR>     <FILE>    <MEM>
 00001012    31'1'    00'.'
 00001022    32'2'    01'.'
 TOP ADDRESS=00000000
 END ADDRESS=00003FFF
:
```

2. To verify an S-type load module against the memory contents in local mode:

```
:V ;S (RET)
                        ← After entering the VERIFY command, set the host
                          system to output data to the E7000.
 TOP ADDRESS=00000000
 END ADDRESS=00003042
:
```

9.4.7	**INTFC_LOAD** **IL**	**Loads program from host system** **— Remote mode**

Command Format

- Load :INTFC_LOAD[Δ<offset>][;[<load module type>][ΔN][ΔDEL]]:<file name> (RET)

<offset>: Value to be added to the load module address (can only be specified for an S-type or HEX-type load module) or the start address (can only be specified for memory image file).
Default: H'0

<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
Default:	SYSROF-type load module

N: Specifies that <line number symbol> is not to be loaded. If omitted, <line number symbol> is loaded.

DEL: Deletes all the symbols before loading. If omitted, does not delete any symbols.

<file name>: Specifies a file name in the host system.

Description

- Load

— Loads a user program into user memory from the host system connected in remote mode. Use the H series interface software for the host system to open the specified file and transfers its contents to the E7000.

```
: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>
```

— If the load module is either S-type or HEX-type, an offset (value to be added) can be specified for the load module address.

```
: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>
```

— Information for symbolic debugging is included in a SYSROF-type load module. When a load module in SYSROF-type format is loaded, unit names of symbols to be defined can be selected as follows:

```
:INTFC_LOAD;R:<file name> (RET)
 ALL SYMBOL LOAD (Y/N)?  x (RET)  ............................. (a)
 LOAD UNIT NAME (name/.)?  <unit name> (RET)  .......... (b)
    .       .       .       .           .
    .       .       .       .           .
    .       .       .       .           .
 LOAD UNIT NAME (name/.)?  . (RET)  ............................. (b)
```

(a) Specifies whether all symbols are to be loaded or symbols are to be selected.

Y: Loads all symbols.

N: Enables the selection of symbols by unit name.

If Y is entered, all symbols are loaded and the confirmation request messages (b) are not displayed. If N is entered, the confirmation request messages are displayed.

(b) Specifies symbol unit name to be defined.

Loading starts when the period (.) is entered.

Up to ten unit names can be defined.

— If the N option is specified, <line number symbol> information among debugging information for the SYSROF-type load module is not loaded.

— Specifying the DEL option deletes all currently defined symbols.

```
:INTFC_LOAD ; R DEL [:<command transferred to host system>] (RET)
```

However, if an attempt is made to delete all symbols including one currently in use (one used and specified with another command), the emulator displays the following message:

```
*** 30:SYMBOL IN USE
```

and terminates command execution without symbol deletion.

Notes

1. The load module cannot be loaded to the areas other than CS0 to CS3 areas and cache data forced read/write area.

2. Verification is not performed. The program must be verified with the INTFC_VERIFY command if necessary.

3. The LOAD command reloads existing symbols to enhance throughput without checking for double definitions. When reloading the same load module, specify the DELoption or delete existing symbols before performing the INTFC_LOAD.

Examples

1. To load a SYSROF-type load module F11.ABS. Symbol information for unit un001 is loaded:

```
:IL :F11.ABS (RET)
 ALL SYMBOL LOAD (Y/N) ?  N (RET)
 LOAD UNIT NAME (name/.) ? un001 (RET)
 LOAD UNIT NAME (name/.) ? . (RET)

 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

2. To load an S-type load module ST.MOT:

```
:IL;S :ST.MOT(RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

9.4.8	INTFC_SAVE IS	Saves program in host system — Remote mode

Command Format

- Save

```
:INTFC_SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
                    [;[<load module type>][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 LF (H'0A) to the end of each record.
<file name>: File name in the host system.

Description

- Save

— Saves the specified memory contents in the specified load module type file in the host system connected in remote mode. Use the H series interface software for the host system. An S-type or HEX-type load module can be saved. SYSROF-type and M-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 option LF is specified, the E7000 adds an LF code (H'0A) to the end of each record in addition to a CR code (H'0D) in the S- or HEX-type load module.

Notes

1. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be saved.

2. Verification is not performed. The program must be verified with the INTFC_VERIFY command if necessary.

Example

To save memory contents in the address range from H'7000 to H'7FFF in the host system file F11.MOT in S-type load module format:

```
:IS 7000 7FFF :F11.MOT (RET)
 TOP ADDRESS=00007000
 END ADDRESS=00007FFF
:
```

9.4.9	INTFC_TRANSFER IT	Transfers file to and from host system — Remote mode

Command Format

- Transfer :INTFC_TRANSFER <file name>[[(S/R)][ΔC]:<host system file name> (RET)

<file name>:	Name of file on floppy disk in the E7000
S:	Transfer from the E7000 to the host system
R:	Reception from the host system (default)
C:	Inserts CR codes before LF codes in files transferred from the host system and removes CR codes from files transferred to the host system
<host system file name>:	File name in the host system

Description

- Transmission

— Transfers only text files from the E7000 to the host system connected in remote mode. Use the H series interface software for the host system.

```
:INTFC_TRANSFER <file name>;S :<host system file name> (RET)
```

- Reception

— Transfers a file from the host system connected in remote mode to the E7000. Only text files can be transferred.

```
:INTFC_TRANSFER <file name>;R :<host system file name> (RET)
```

If the specified file already exists, the message below is displayed. Enter Y or N.

```
OVERWRITE (Y/N)  ? (a) (RET)
```

(a) Y: Overwrites the existing file with the new file.
N: Aborts the command.

— The E7000 can receive only text files (ASCII characters) that can be displayed. If the E7000 receives another type of data, an error is generated and the command is aborted.

This command is terminated with EOF (H'1A).

— With a UNIX-based host system, each record is terminated with a single LF code and no CR code. To receive such records, specify option C.

Examples

1. To transfer file SAMPLE.S from the E7000 to the host system file SAM.S in remote mode:

```
:IT SAMPLE.S ;S :SAM.S (RET)
:
```

2. To transfer the host system file FILE.TXT to the E7000 file F.T:

```
:IT F.T ;R : FILE.TXT (RET)
:
```

9.4.10	**INTFC_VERIFY** **IV**	**Verifies memory contents against host system file — Remote mode**

Command Format

- Verification :VERIFY [Δ<offset>][;<load module type>]:<file name> (RET)

<offset>: Value to be added to the address (can only be specified for a S-type or HEX-type load module) or the start address (can only be specified for memory image file).
Default: H'0
<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
Default: SYSROF-type load module
<file name>: File name in the host system

Description

- Verification

— Verifies data transferred from the host system connected in remote mode against data in memory. Use the H series interface software for the host system.

```
:INTFC_VERIFY[;<load module type>]:<file name> (RET)
```

— If a verification error occurs, verification terminates immediately and the address and its contents are displayed as follows. Note that only one verification error can be detected and its contents are displayed.

```
<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)

— If the load module is either S-type or HEX-type, an address offset (value to be added or subtracted) of the load module can be specified.

```
:INTFC_VERIFY<offset>; <load module type> [:<file name>] (RET)
```

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

Notes

1. Symbolic information cannot be verified.

2. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be verified.

Example

To verify a SYSROF-type load module F1.ABS against the memory contents:

```
:IV :F1.ABS  (RET)
 <ADDR>     <FILE>    <MEM>
 00001012    31'1'     00'.'
 TOP ADDRESS=00000000
 END ADDRESS=00003FFF
:
```

Section 10 Data Transfer from Host System Connected by LAN Interface

10.1 Overview

The optional LAN board supports the FTP client function. This function enables the following data transfer between the E7000 and the host system connected through the LAN interface.

- Loads the load module file from the host system into the user system memory
- Saves the user system memory contents into a host system file
- Transfers files between the E7000 and the host system

The E7000 supports the LAN commands listed in table 10.1 to transfer data between the E7000 and the host system. These commands are explained in section 10.3, LAN Commands.

These commands cannot be used in the E7000PC.

Table 10.1 LAN Commands

Command	Function	Usable/Unusable in Parallel Mode
ASC	Specifies the file type as ASCII	Usable
BIN	Specifies the file type as binary	Usable
BYE	Terminates the FTP interface (Re-connects the FTP interface with the FTP command)	Usable
CD	Modifies the file directory name of the FTP server	Usable
CLOSE	Disconnects the host system from the FTP interface (Re-connects the host system to the FTP interface with the OPEN command)	Usable
FTP	Connects the host system and E7000 via the FTP interface	Usable
LAN	Displays E7000 IP address	Usable
LAN_HOST	Specifies, modifies, and displays the name and IP address of the host to be connected via the FTP command	Unusable
LAN_LOAD	Loads a load module file from the host system to memory via the FTP interface	Unusable
LAN_SAVE	Saves the specified memory contents in the LAN host system connected via the FTP interface	Unusable
LAN_TRANSFER	Transfers a file between the host system and E7000	Unusable
LAN_VERIFY	Verifies memory contents against the host system file	Unusable
LS	Displays the host system directory connected via the FTP interface	Usable
OPEN	Connects the host system to the FTP interface	Usable
PWD	Displays the current directory name of the host system to be connected via the FTP interface	Usable
STA	Displays the type of a file to be transferred	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 E7000 is connected to the host system through Telnet, the E7000 can be disconnected from the Telnet with the LOGOUT command. For details on the Telnet interface, refer to section 3.4.1, Power-on Procedure for LAN Interface, in Part I, E7000 Guide. Note that the FTP can be connected via Telnet or RS-232C.

10.2 LAN Data Transfer

10.2.1 Setting the Data Transfer Environment

The optional LAN board enables the data transfer between the E7000 and host system via 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 system environment, including the host system name and IP address, to the network database of the host system. If the operating system of the host system is UNIX, the host system environment is specified in the /etc/hosts file. For details, refer to the appropriate host system's User’s Manual.

2. Specify the following E7000 environment:

   - E7000 IP address

     Specify the E7000 IP address with the E7000 monitor's L command. Since the E7000 IP address is written to the EEPROM, it must be written only once. The E7000 IP address can be modified as required.

   - Host system IP address (host system connected via FTP interface)

     Specify the name and IP address of the host system to be connected to the E7000 via the FTP interface when initiating the E7000 system program. For details, refer to section 10.3.8, LAN_HOST. The specified host name and IP address is written to the system disk. Accordingly, the E7000 is automatically connected to the host system simply by initiating the system disk. The host system name and IP address can be modified as required.

10.2.2 Data Transfer

Data transfer is performed by connecting the E7000 to the host system via the FTP interface after the environmental settings have been completed. In the FTP interface, the optional LAN board supports only the client function. Note that the E7000 and host system must be connected to the FTP interface in that order. Transfer data using the following procedure.

Procedure:

1. Initiate the E7000 system program by using the system disk, on which the host system name and IP address have been defined by the LAN_HOST command.

2. Connect the E7000 to the designated host system with the FTP command using the format shown below. Enter the host system name defined by the LAN_HOST command. In addition, enter the user name and password.

   ```
   :FTP <host system name> (RET)
    Username <user name> (RET)
    Password <password> (RET)
    login command success
   FTP>
   ```

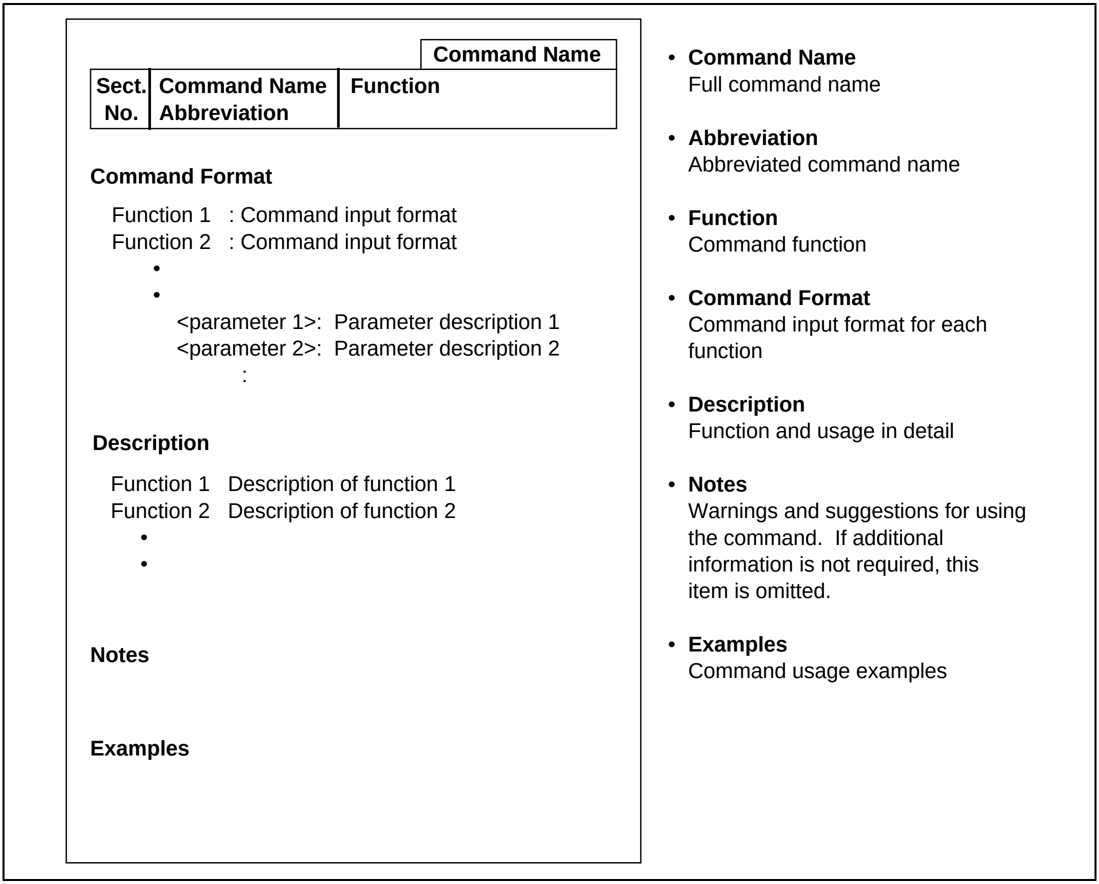
3. Transfer data using the LAN_LOAD, LAN_SAVE, LAN_VERIFY, or LAN_TRANSFER command after the FTP interface is established. For details, refer to the corresponding command descriptions.

10.2.3 Notes on FTP Interface

Before turning off the E7000 main power, the FTP interface must be terminated using the BYE command. Otherwise, the host system interface processing may remain uncompleted. In this case, the FTP interface cannot be re-established correctly even if the E7000 is re-initiated.

10.3 LAN Commands

This section provides details of LAN commands in the format shown in figure 10.1:



Sect. No.	Command Name Abbreviation	Function

Command Name

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
- **Abbreviation**
  Abbreviated command name
- **Function**
  Command function
- **Command Format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and suggestions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 10.1 Format of LAN Command Description

Symbols used in the command format have the following meanings:

[ ]: Parameters enclosed by [ ] can be omitted.
(a/b): One of the number of parameters enclosed by ( ) and separated by /, that is, either a or b must be specified.
< >: Contents shown in < > are to be specified or are 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 underlining is used throughout this manual to indicate input, it is not used in the command format sections of these descriptions.

10.3.1	**ASC** **ASC**	**Specifies the file type as ASCII**

Command Format

- Setting : ASC (RET)

Description

- Setting

  Specifies a file type as ASCII in the FTP interface. This specification is required to transfer text files with the LAN_TRANSFER command. Before transferring the command chain file created by the host system, specify the ASCII type with this command. To load a 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>
```

10.3.2	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_VERIFY, LAN_SAVE, or LAN_TRANSFER command. To load or verify a SYSROF-type load module file, binary must be specified with this command. Otherwise, a transfer error will occur. At E7000 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>
```

10.3.3	**BYE** **BYE**	**Terminates the FTP interface**

Command Format

- Termination : BYE (RET)

Description

- Termination

  Terminates the FTP interface and changes the prompt to a colon (:). To re-establish the FTP interface, re-enter the FTP command.

Example

To terminate the FTP interface:

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

10.3.4	**CD** **CD**	**Modifies the file directory of the FTP server**

Command Format

- Modification : CD Δ<directory name> (RET)

  <directory name>: Name of directory to be modified

Description

- Modification

  Changes the current directory of the FTP server to the specified directory. The modified directory must be formatted depending on which host system is connected via the FTP interface.

Example

To change the current directory of the FTP server to the specified directory:

```
FTP> CD subdir (RET)
 cd command success
FTP>
```

10.3.5	**CLOSE** **CLOSE**	**Disconnects the host system from the FTP interface**

Command Format

- Disconnection : CLOSE (RET)

Description

- Disconnection

  Disconnects the FTP interface from the host system to which it is currently connected. Before changing host systems, disconnect the FTP interface with this command and re-connect with the OPEN command.

Example

To disconnect the FTP interface and change the host system to be connected:

```
FTP> CLOSE (RET)
 bye command success
FTP> OPEN HOST1 (RET)
 username ABC (RET)
 password ****** (RET)
 login command success
FTP>
```

10.3.6	**FTP** **FTP**	**Connects host system and E7000 via the FTP interface**

Command Format

- Connection : FTP <host name> (RET)

  <host name>: Name of the LAN host system to be connected with the FTP server

Description

- Connection

  — Connects the host system and E7000 via the FTP interface to enable data transfer with the LAN_LOAD, LAN_SAVE, LAN_VERIFY, or LAN_TRANSFER command. The host name specified in this command must be defined with the LAN_HOST command.

  — If <host name> matches the host name specified with the LAN_HOST command, enter the user name and password in the following format. After the FTP command execution, a prompt changes from a colon (:) to FTP>. In this case, emulation commands and floppy disk utility commands can be executed.

```
: 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 system can be connected via the FTP. For a host system that can login by using only user name, use a login format that requires a password.

FTP

Example

To connect the E7000 to host system HOST1 via the FTP interface:

```
:FTP HOST1 (RET)
  Username USER1 (RET)
  Password ******** (RET)
  login command success
FTP>
```

10.3.7	LAN LAN	Displays E7000 IP address

Command Format

- Display : LAN (RET)

Description

- Display

  — Displays the E7000’s internet (IP) address stored in the EEPROM, which is incorporated in the emulator station, in the following format:

```
: LAN    (RET)
  E7000 INTERNET ADDRESS xxx.xxx.xxx.xxx
                                  (a)
```

  (a): E7000 IP address stored in the EEPROM

  — Specify the IP address with the E7000 monitor command L.

Example

To display the E7000 IP address:

```
:LAN (RET)
 E7000 INTERNET ADDRESS 128.1.1.10
:
```

10.3.8	LAN_HOST LH	**Specifies, modifies, and displays the name and IP address of the host to be connected by the FTP command**

Command Format

- Specification and modification : LAN_HOST;S (RET)
- Display : LAN_HOST (RET)

Description

- Specification

— Specifies the name and internet (IP) address of the host system to be opened with the FTP command. A maximum of nine names and internet addresses can be specified.

— The specified host name and IP address can be modified in interactive mode as shown below. After displaying the specified host names and internet addresses, the E7000 waits for the selection number input. Note that new data is written to the E7000 system disk; insert the system disk before executing this command.

```
: LAN_HOST; S   (RET)
  NO   <HOST NAME>   <IP ADDRESS>   NO   <HOST NAME>   <IP ADDRESS>
  01     xxxxxx         xx.xx.xx.xx    02     xxxxxx          xx.xx.xx.xx      (a)
  03     xxxxxx         xx.xx.xx.xx    04     xxxxxx          xx.xx.xx.xx
  05     xxxxxx         xx.xx.xx.xx    06     xxxxxx          xx.xx.xx.xx
  07     xxxxxx         xx.xx.xx.xx    08     xxxxxx          xx.xx.xx.xx
  09     xxxxxx         xx.xx.xx.xx
  PLEASE SELECT NO ?   1  (RET)                                              (b)
  01     HOST NAME       xxxxxx          ?   xxxxxx   (RET)                  (c)
  01     IP ADDRESS      xx.xx.xx.xx     ?   xx.xx.xx.xx   (RET)             (d)
  PLEASE SELECT NO ?   . (RET)                                               (e)
```

(a) Displays host system name and IP address currently defined. If nothing is specified, displays a space. NO indicates selection number.

(b) Enters the selection number (1–9) to be set or modified.

(c) Displays the host system name for the specified selection number. To specify a new host system name, enter the host system name using six or less characters. To delete the old host system name, enter –(RET).

(d) Displays the IP address for the specified selection number. Enter a new IP address in decimal format.

Example: 128.1.1.16

(e) Indicates the selection number input wait state. To specify or modify another host system name or IP address, repeat steps (a) to (d). To terminate this command, enter a period (.) and hit the (RET) key. The following confirmation message is displayed:

```
PLEASE SELECT NO ?      . (RET)
OVERWRITE (Y/N)      x (RET)
```

x: Enter Y to write new data in the system disk; enter N to terminate the command without storing the new data.

If an equal (=) and the (RET) keys are entered instead of the period and (RET) keys, the current host system name and IP address settings are displayed.

— Specified host system names and internet addresses are stored in the LANCNF.SYS file of the E7000 system disk. After storing data in the LANCNF.SYS file, the E7000 system program is terminated, along with the Telnet interface. To use the E7000 via the Telnet interface, re-initiate the E7000 and connect the E7000 to the Telnet from the host system.

— Before executing the FTP command, specify the name and internet address of the host system to be connected with the FTP command. If the E7000 is initiated by the system disk where the name and IP address of the host system is defined, the name and IP address of the host system need not be specified.

- Display

Displays the LAN host system names and internet addresses specified in the LANCNF.SYS file of the E7000 system disk.

```
:LAN_HOST (RET)
```

Examples

1. To add a host system to be connected via the Telnet interface:

```
:LH; S (RET)
 NO  <HOST NAME>   <IP ADDRESS>   NO  <HOST NAME>   <IP ADDRESS>
 01   H0ST1         128.1.1.1     02   HST2          128.1.1.4
 03                               04
 05                               06
 07                               08
 09

PLEASE SELECT NO ? 3 (RET)   ----- (New host system is defined as No.3)
03   HOST NAME             ?   HOSTX (RET)
03   IP ADDRESS            ?   128.1.1.8 (RET)
PLEASE SELECT NO  ?   . (RET)
OVERWRITE (Y/N)   Y (RET)

START E7000
 S:START E7000
 R:RELOAD & START E7000
 B:BACKUP FD
 F:FORMAT FD
 L:SET LAN PARAMETER
 T:START DIAGNOSTIC TEST
  (S/R/B/F/L/T)  ? S (RET) --------- (The E7000 is restarted with S)
```

2. To display all of the defined host system names and IP addresses:

```
:LH  (RET)
 NO  <HOST NAME>   <IP ADDRESS>   NO  <HOST NAME>   <IP ADDRESS>
 01   HOST1         128.1.1.1     02   HST2          128.1.1.4
 03   HOSTX         128.1.1.8     04
 05                               06
 07                               08
 09
:
```

10.3.9	**LAN_LOAD** **LL**	**Loads a load module file from the host system to memory via the FTP interface**

Command Format

- Load : LAN_LOAD [Δ<offset>][;[<load module type>][ΔN][ΔDEL]]:<file name> (RET)

<offset>: Value to be added to the load module address (can only be specified for an S-type or HEX-type load module) or the start address (can only be specified for M-type load module).
Default: H'0

<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
Default: SYSROF-type load module

N: Specifies that <line number symbol> is not to be loaded. If omitted, <line number symbol> information is loaded.

DEL: Deletes all the symbols before loading. If omitted, does not delete any symbols.

<file name>: A LAN host system file name

Description

- Load

— Loads a load module file from the host system to memory via the FTP interface. Before executing this command, the E7000 must be connected to the host system with the FTP command.

— The current load address is displayed in the format below.

```
LOADING ADDRESS = xxxxxxxx
                     (a)
```

(a) The current load address display is continuously updated

— When loading is completed, the start and end addresses are displayed as follows:

```
TOP ADDRESS     = <start address>
END ADDRESS     = <end address>
```

— If the load module is either S-type or HEX-type, an offset (value to be added) can be specified for the load module address.

```
: LAN_LOAD  <offset>; S:<file name>    (RET)
```

If an offset is specified, the load address is calculated by adding the offset to the address in the load module.

— Information for symbolic debugging is included in a SYSROF-type load module. Unit name in a SYSROF-type load module can be selected and loaded in SYSROF units.

If a SYSROF-type load module is specified, the following message is displayed to prompt the input of symbol units:

```
: LAN_LOAD  :<file name>    (RET)
  ALL SYMBOL LOAD (Y/N)   ?  x (RET)   -------------------------------------- (a)
  LOAD UNIT NAME (name/.) ? <unit name> (RET) --------------------------- (b)
     .          .              .                        .
     .          .              .                        .
 LOAD UNIT NAME (name/.) ?   .  (RET) --------------------------------------- (b)
```

(a) Specifies whether all symbols are to be loaded or are to be selected.

Y: Loads all symbols.

N: Enables the selection of symbols by unit names.

If Y is entered, all symbols are loaded and the confirmation request messages (b) are not displayed. If N is entered, the confirmation request messages are displayed.

(b) Specifies a unit name. Loading starts when the period (.) is entered as a response to a confirmation request message. Up to 10 unit names can be specified.

— If option N is specified, <line number symbol> information among the symbol information for the module is not loaded.

— Specifying the DEL option deletes all currently defined symbols.

```
:LAN_LOAD ; R DEL [:<command transferred to host system>] (RET)
```

However, if an attempt is made to delete all the symbols including one currently in use (one used and specified with another command), the emulator displays the following message:

```
*** 30:SYMBOL IN USE
```

and terminates command execution without symbol deletion.

Notes

1. A load module file cannot be loaded to the areas other than CS0 to CS3 areas and cache data forced read/write area.

2. Verification is not performed. The program must be verified with the LAN_VERIFY command, if necessary.

3. The LAN_LOAD command reloads existing symbols to enhance throughput, without checking for double definitions. When reloading the same load module, specify the DEL option or delete the symbols before performing the LAN_LOAD.

4. Before loading a SYSROF-type load module, the file contents must be converted into binary code with the BIN command. At E7000 initiation, binary code is selected as the default. However, if ASCII is selected with the ASC command, change the file contents to binary code with the BIN command before loading.

Example:

To load a SYSROF-type load module, enter the following command line. F11.ABS indicates the host system file name. Before entering the LL command, connect the E7000 to the host system with the FTP command:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)

FTP> LL :F11.ABS (RET)
 ALL SYMBOL LOAD (Y/N) ? N (RET)
 LOAD UNIT NAME (name/.) ? un0001 (RET)
 LOAD UNIT NAME (name/.) ? . (RET)

 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
FTP>
```

10.3.10	LAN_SAVE LSV	Saves the specified memory contents in the LAN host system connected via the FTP interface

Command Format

- Save : LAN_SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
  [;[<load module type>][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:	LF (H'0A) is added to the end of each record.	
<file name>:	A LAN host system file name	

Description

- Save

— Saves the specified memory contents in the host system connected via the FTP interface. Either an S-type or HEX-type load module can be saved. A SYSROF-type load module cannot be saved. Before executing this command, connect the E7000 to the host system with the FTP command.

— The current save address is displayed as follows:

```
SAVING ADDRESS = xxxxxxxx
                    (a)
```

(a) Current save address display is 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 LF (H'0A) code as well as CR (H'0D) code is added to the end of each S- or HEX-type load-module record.

Notes

1. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be saved.

2. Verification is not performed. Verify the program with the LAN_VERIFY command, if necessary.

Example

To save the memory contents in the addresses from H'7000 to H'7FFF in the host system as a S-type load module file (file name: F11.S), enter the following command line. Before entering the LSV command, connect the E7000 to the host system with the FTP command:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>LSV 7000 7FFF :F11.S (RET)

 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
FTP>
```

10.3.11	**LAN_TRANSFER** **LTR**	**Transfers a file between the host system and E7000**

Command Format

- Transfer : LAN_TRANSFER <file name1>[;[(S/R)]]:<file name2> (RET)

<file name1>: Name of file on floppy disk in the E7000
S: Transfer from the E7000 to the host system
R: Receive from the host system (Default at E7000 initiation)
<file name 2>: A host system file name

Description

- Transfer

Performs file transmission and reception between the E7000 and host system via the FTP interface. Before entering this command, the following steps must be completed.

(1) The E7000 must be connected to the host system with the FTP command.

(2) The file type must be specified as either binary or ASCII with the BIN or ASC command, respectively.

— Transmission

Transfers files from the E7000 to the LAN host system via the FTP interface.

```
: LAN_TRANSFER <file name1> ; S :<file name2> (RET)
```

— Reception

Transfers files from the host system to the E7000 via the FTP interface. Before transferring a command file for the COMMAND_CHAIN command, specify the file type as ASCII with the ASC command.

```
: LAN_TRANSFER <file name1> ; R :<file name2> (RET)
```

If the specified file already exists, the message below is displayed. Enter Y or N.

```
OVERWRITE (Y/N) ?     (a)  (RET)
```

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

Examples:

1. To transfer file SAMPLE.S from the E7000 to file TEST.S on the host system, enter the following command line. Before executing the LTR command, the E7000 must be connected to the host system with the FTP command:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>LTR SAMPLE.S ;S :TEST.S (RET)
FTP>
```

2. To transfer ASCII file COM.CC from the host system to file E7.CC on the E7000, enter the following command line. Before executing the LTR command, the file type must be specified as ASCII with the ASC command:

```
FTP>ASC (RET)
 asc command success
FTP>LTR E7.CC ;R :COM.CC (RET)
FTP>
```

10.3.12	**LAN_VERIFY** **LV**	**Verifies memory contents against the host system file**

Command Format

- Verification : LAN_VERIFY [Δ<offset>][;<load module type>]:<file name> (RET)

<offset>: Value to be added to the address (can be specified only for an S-type or HEX-type load module) or the start address (can only be specified for memory image file).
Default: H'0

<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
Default: SYSROF-type load module

<file name>: A LAN host system file name

Description

- Verification

— Verifies file transferred from the host system connected via the FTP interface against data in memory in the following format. Before executing this command, connect the E7000 to the host system 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)

— If the load module is either S-type or HEX-type, an address offset (to be added or subtracted) of the load module can be specified for the load module address. For subtraction, put a – in front of the value.

FTP > LAN_VERIFY <offset> ; S [:<file name>] (RET)

If an offset is specified, the load address is calculated by adding the offset to the address in the load module.

Notes

1. Symbolic information cannot be verified.

2. Data in the areas other than CS0 to CS3 areas and cache data forced read/write area cannot be verified.

3. Before verifying a SYSROF-type load module, the file contents must be converted into binary code with the BIN command. At E7000 initiation, binary code is selected as default. However, if ASCII is selected with the ASC command, change file contents to binary code with the BIN command before verifying.

Example

To verify a SYSROF-type load module file F11.ABS with the memory:

```
:FTP HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>LV :F11.ABS (RET)
FTP>
```

10.3.13	**LS** **LS**	**Displays the host system directory connected via the FTP interface**

Command Format

- Display : LS [Δ <directory name>] (RET)

  <directory name>: Name of host system directory
  (Default: Current directory of the host system)

Description

- Display

  Displays the specified directory in the host system connected via the FTP interface. If <directory name> is omitted, the current directory contents is displayed. Note that the directory name must be specified according to the connected host system format.

Example

To display the contents of current directory of the host system:

```
FTP>LS (RET)
 abc.s
 xyz
FTP>
```

10.3.14	**OPEN** **OPEN**	**Connects the host system to the FTP interface**

Command Format

- Connection : OPEN <host system name> (RET)

  <host system name>: Name of host system to be connected via the FTP interface
  (The host system name must have been defined with the LAN_HOST command)

Description

- Connection

  Connects the E7000 to the specified host system via the FTP interface. This command can also be used to change the host system to be connected to the E7000. To change the host system correctly, first disconnect the current host system by using the CLOSE command and then connect the new host system by using this command.

```
FTP>OPEN <host system 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 system can be connected via the FTP. When the host system requires only the user name to login, use a login format that requires a password.

Example

To disconnect the E7000 from the current host system and connect it to the new host system HOST1:

```
FTP>CLOSE (RET)
 bye command success
FTP>OPEN HOST1 (RET)
 Username USER1 (RET)
 Password ******** (RET)
 login command success
FTP>
```

10.3.15	**PWD** **PWD**	**Displays the current directory name of the host system connected via the FTP interface**

Command Format

- Display : PWD (RET)

Description

- Display

  Displays the current directory name of the host system connected via the FTP interface.

Example

To display the current directory name of the host system connected via the FTP interface:

```
FTP>PWD (RET)
 /usr/e7000
FTP>
```

10.3.16	**STA** **STA**	**Displays the type of a file 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_VERIFY, LAN_SAVE, or LAN_TRANSFER command.

```
FTP>STA (RET)
 type mode is BINARY      (Binary)

FTP>STA (RET)
 type mode is ASCII       (ASCII)
```

Example

To display the type of file to be transferred:

```
FTP>STA (RET)
 type mode is BINARY
FTP>
```

10.3.17	LOGOUT LO	Disconnects from the Telnet

Command Format

- Disconnection : LOGOUT (RET)

Description

- Disconnection

  Disconnects the E7000 from the Telnet. This command is valid only when the E7000 is connected to the host system via the Telnet interface.

Example

To disconnect the E7000 from the Telnet interface:

```
 :LO (RET)
```

Section 11 Data Transfer between E7000PC and IBM PC

11.1 Overview

The following data transfers between the E7000PC and a host system (IBM PC) can be performed by the commands listed in table 11.1.

- Loads a load module file in the host system to the memory on the user system.
- Saves data in the user system memory as a load module file in the host system.
- Performs a file transfer between E7000PC and IBM PC.

Table 11.1 E7000PC-Related Data Transfer Commands

Command Name	E7000PC Command Use
LOAD	Loads program from host system (IBM PC)
SAVE	Saves program in host system (IBM PC)
VERIFY	Verifies memory contents against host system file (IBM PC)

11.2 E7000PC and IBM PC System Connection

The flow of data transfer between the E7000PC and IBM PC is shown in figure 11.1.

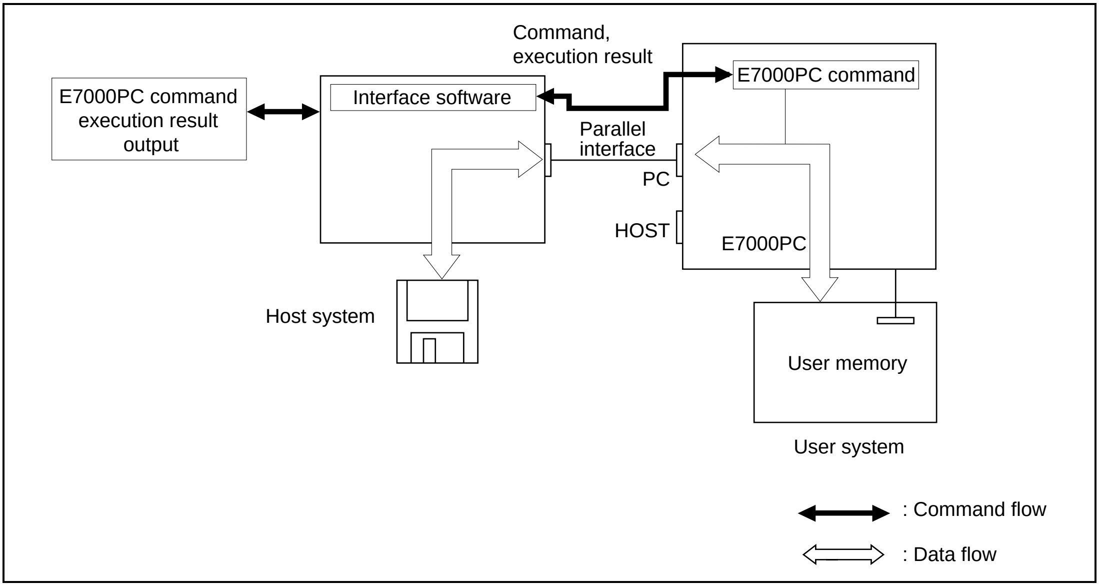


Figure 11.1 Flow of Data Transfer

Procedure:

Start up host system

Start up interface software

The H-series interface software start-up message is displayed:

```
H-SERIES INTERFACE (type no.) Ver n.m
Copyright (C) Hitachi, Ltd. 1993
Licensed Material of Hitachi, Ltd.
INTERFACE BOARD ADDRESS = yyyy:zzzz,
TERMINATE CODE = tt
```

Start up the E7000PC

The E7000PC start-up message is displayed on the host system. Emulator commands can now be entered from the host system.

<Data transfer from IBM PC to E7000PC>
Execute E7000PC data receive command

The E7000PC data receive command (LOAD or VERIFY) can transfer data from the host system to the E7000PC.

Example:
LOAD:<host system file name>

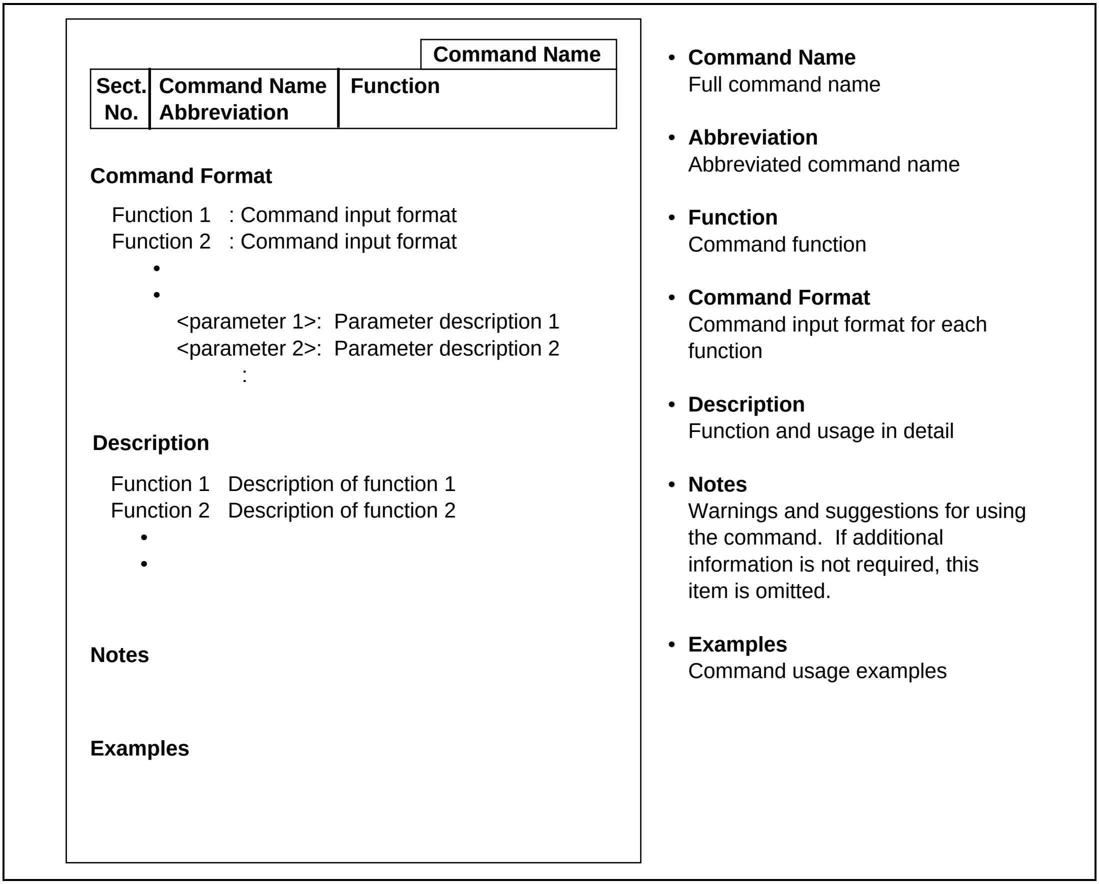
<Data transfer from E7000PC to host system>
Execute E7000PC data transmission command

The E7000PC data transmission command (SAVE) can transfer data from the E7000PC to the host system.

Example:
SAVE 0 1FFF:<host system file name>

11.3 E7000PC-Related Data Transfer Commands

This section provides details of host-system related commands using the format shown in figure 11.2.



Sect. No.	Command Name Abbreviation	Function

Command Name

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
- **Abbreviation**
  Abbreviated command name
- **Function**
  Command function
- **Command Format**
  Command input format for each function
- **Description**
  Function and usage in detail
- **Notes**
  Warnings and suggestions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 11.2 Format of E7000PC-Related Data Transfer Command Description

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 are displayed.
...: The entry specified just before this symbol can be repeated.
Δ: Indicates a space. Used only for command format description.
(RET): Indicates pressing the (RET) key.

Although underlining is used throughout this manual to indicate input, it is not used in the command format parts of these descriptions.

11.3.1	**LOAD** **L**	**Loads program from host system (IBM PC)**

Command Format

- Load :LOAD[Δ<offset>][;[<load module type>][ΔN][ΔDEL]]:<file name> (RET)

<offset>: Value to be added to the address (can be specified only for an S-type or HEX-type load module) or the start address (can only be specified for memory image file).
Default: H'0
<load module type>: Load module type
S: S-type load module
H: HEX-type load module
M: Memory image file
R: SYSROF-type load module
Default: SYSROF-type load module
N: Specifies that <line number symbol> is not to be loaded. If omitted, <line number symbol> is loaded.
DEL: Deletes all the symbols before loading. If omitted, does not delete any symbols.
<file name>: Specifies a file name in the host system (IBM PC).

Description

- Load

— Loads a user program into user memory from the host system. Use the H-series interface software for the host system to open the specified file and transfers its contents to the E7000PC.

```
: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>
```

— If the load module is either S-type or HEX-type, an offset (value to be added) can be specified for the load module address.

```
:LOAD <offset>;S:<file name> (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

— Information for symbolic debugging is included in a SYSROF-type load module. When a load module in SYSROF-type format is loaded, unit names of symbols to be defined can be selected as follows:

```
:L;R:<file name> (RET)
 ALL SYMBOL LOAD (Y/N)?  x (RET)  ..................................(a)
 LOAD UNIT NAME (name/.)?  <unit name> (RET)  ...............(b)
    .        .        .        .              .
    .        .        .        .              .
    .        .        .        .              .
 LOAD UNIT NAME (name/.)?  . (RET)  ..................................(b)
```

(a) Specifies whether all symbols are to be loaded or symbols are to be selected.

Y: Loads all symbols.

N: Enables the selection of symbols by unit name.

If Y is entered, all symbols are loaded and the confirmation request messages (b) are not displayed. If N is entered, the confirmation request messages are displayed.

(b) Symbol unit name to be defined

Loading starts when the period (.) is entered.

Up to ten unit names can be defined.

— If the N option is specified, <line number symbol> debugging information for the SYSROF-type load module is not loaded.

— Specifying the DEL option deletes all currently defined symbols.

```
:LOAD ; R DEL [:<command transferred to host system>] (RET)
```

However, if an attempt is made to delete all the symbols including one currently in use (one used and specified with another command), the emulator displays the following message:

```
*** 30:SYMBOL IN USE
```

and terminates command execution without symbol deletion.

Notes

1. The load module cannot be loaded to areas other than the CS0 to CS3 areas and the cache data forced read/write area.

2. Verification is not performed. The program must be verified with the VERIFY command if necessary.

3. The LOAD command reloads existing symbols to enhance throughput without checking for double definitions. When reloading the same load module, specify the DEL option or delete existing symbols before performing the LOAD.

Examples

1. To load a SYSROF-type load module F11.ABS. Symbol information for unit un001 is loaded:

```
:L :F11.ABS (RET)
 ALL SYMBOL LOAD (Y/N) ?  N (RET)
 LOAD UNIT NAME (name/.) ? un001 (RET)
 LOAD UNIT NAME (name/.) ? . (RET)

 TOP ADDRESS = 00007000
 END ADDRESS = 00007A3F
:
```

2. To load an S-type load module ST.MOT:

```
:L ;S :ST.MOT(RET)
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

11.3.2	SAVE SV	Saves program in host system (IBM PC)

Command Format

- Save :SAVEΔ<start address>(Δ<end address>/Δ@<number of bytes>)
  [;[<load module type>][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 LF (H'0A) to the end of each record.
<file name>: File name in the host system (IBM PC).

Description

- Save

— Saves the specified memory contents in the specified load module type file in the host system. Use the H-series interface software for the host system. An S-type or HEX-type load module can be saved. A SYSROF-type and M-type load module cannot be saved.

```
: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 option LF is specified, the E7000PC adds an LF code (H'0A) to the end of each record in addition to a CR code (H'0D) in the S- or HEX-type load module.

Notes

1. Data in areas other than the CS0 to CS3 areas and the cache data forced read/write area cannot be saved.

2. Verification is not performed. Verify the program with the VERIFY command if necessary.

Example

To save memory contents in the address range from H'7000 to H'7FFF in the host system file F11.MOT in S-type load module format:

```
:SV 7000 7FFF :F11.MOT (RET)
 TOP ADDRESS=00007000
 END ADDRESS=00007FFF
:
```

11.3.3	VERIFY V	Verifies memory contents against host system file (IBM PC)

Command Format

- Verification   :VERIFY [Δ<offset>][;<load module type>]:<file name> (RET)

<offset>: Value to be added to the address (can be specified only for an S-type or HEX-type load module) or the start address (can only be specified for memory image file).
Default: H'0

<load module type>: Load module type

S:	S-type load module
H:	HEX-type load module
M:	Memory image file
R:	SYSROF-type load module
Default:	SYSROF-type load module

<file name>: File name in the host system

Description

- Verification

— Verifies data transferred from the host system against data in memory. Use the H-series interface software for the host system.

```
:VERIFY[;<load module type>]:<file name> (RET)
```

— If a verification error occurs, verification terminates immediately and 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)

— If the load module is either S-type or HEX-type, an address offset (value to be added or subtracted) of the load module can be specified.

:VERIFY<offset>;S:<file name in the host system> (RET)

If an offset is specified, a verification address is calculated as follows:

Verification address = <load module address> + <offset>

Notes

1. Symbolic data cannot be verified.

2. Data in areas other than the CS0 to CS3 areas and the cache data forced read/write area cannot be verified.

Example

To verify a SYSROF-type load module F1.ABS against the memory contents:

```
:V :F1.ABS  (RET)
 <ADDR>     <FILE>    <MEM>
 00001012  31'1'     00'.'
 TOP ADDRESS=00000000
 END ADDRESS=00003FFF
:
```

Section 12 Error Messages

12.1 E7000 Error Messages

The emulator system program outputs error messages in the format below. Table 12.1 lists error messages, descriptions of the errors, and error solutions.

```
*** xx:  <error message>
    xx:  Error No.
```

Table 12.1 Emulator Error Messages

Error No.	Error Message	Description and Solution
1	INTERNAL ERROR (nn)	Error occurred in the emulator program or station Error code nn gives specific details. Contact any Hitachi agency and inform them of the code and statement.
2	HOST I/O ERROR (nn)	I/O error occurred between the emulator and host system. Error code nn gives specific details. Refer to table 12.2.
3	FD I/O ERROR (nn)	Error occurred during floppy disk read/write. Error code nn gives specific details. Refer to table 12.3.
5	INVALID EMULATOR POD	The connected emulator pod is not supported by this emulator program or an error occurs in the connection between the emulator pod and emulator. Check the emulator program and emulator pod type numbers, and check the connection between the emulator station and the emulator pod.
6	USER SYSTEM NOT READY	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.
7	PRINTER NOT READY	The printer is not connected or is not turned on. Check the printer power and connection.
8	PAPER EMPTY	The printer is out of paper. Reload paper.
9	FD NOT READY	The floppy disk cannot be read from or written to. Check that a disk is inserted.
10	FD WRITE PROTECT	The floppy disk is write-protected. Remove write protection or use another floppy disk .
11	FD CRC ERROR	CRC error during disk read/write. Reformat or change disks.

Table 12.1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
12	FD UNFORMATTED	The floppy disk is not formatted. Format it or exchange it with a formatted disk.
13	FILE NOT FOUND	The specified file was not found. Check name and disk contents.
14	INVALID FILE NAME	Invalid file name format. Check format specifications.
15	INVALID FILE	The specified file has invalid contents and cannot be read from or written to. Check the contents of the specified file.
16	NOT SAME SIZE	Files to be verified are not of the same size. Check the contents of the specified file.
17	NOT SAME FORMAT	The specified file cannot be read because its format is different. Specify a correct file.
18	FILE TOO LARGE MAX xxxxx BYTES	File to be copied is too large. The maximum size allowed is shown as xxxx. If the symbols are defined, delete the symbols and re-execute the command, or back up the file with the emulator monitor command.
20	SYNTAX ERROR	Command syntax is incorrect. Correct the syntax.
21	INVALID COMMAND	The specified command was not found, or this command cannot be specified during GO command execution 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.
25	SYMBOL NOT FOUND	The specified symbol was not found. Check whether the specified symbol is defined and specify a correct symbol.
26	INVALID SYMBOL	Only a unit name symbol is specified. Specify it with a function and variable names.
27	INVALID CONDITION	Invalid conditions are specified. Correctly enter the conditions.
28	DOUBLE DEFINITION	Item to be registered has already been defined. Delete existing item and re-register.
29	CC COMMAND IN COMMAND FILE	The command file contains a COMMAND_CHAIN command which cannot be used. Delete COMMAND_CHAIN from the file.

Table 12.1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
30	SYMBOL IN USE	The specified symbol cannot be deleted because it has already been used in a BREAK, BREAK_SEQUENCE, BREAK_ CONDITION, TRACE_MEMORY or TRACE_CONDITION command. Clear the symbol in that command, and delete it again.
31	INSUFFICIENT MEMORY	Insufficient memory for command execution. Memory was assigned within the available memory size.
32	INVALID ASM MNEMONIC	An instruction mnemonic in an assembly sentence is invalid. Correct it.
33	INVALID ASM OPERAND	An operand in an assembly statement is invalid. Correct it.
34	ALREADY ASSIGNED	The specified printer or file has already been assigned. Cancel the assignment and re-enter the command.
35	CAN NOT USE THIS MODE	• GO command GO command cannot be executed because the execution mode settings are invalid. Correctly specify the mode. • MOVE_TO _RAM An attempt was made to execute the MOVE_TO_RAM command in single-chip mode. This command cannot be executed in single-chip mode.
36	TOO MANY SYMBOLS	No more symbols can be registered. To load the same program, this error message is displayed because the emulator does not check the symbol double definition. Delete and re-register.
37	TOO MANY POINTS	Too many points are specified. Remove any unnecessary settings and re-enter.
38	SET POINT NOT RAM	A write-protected address is specified by the BREAK or BREAK_SEQUENCE command. Specify a correct address.
39	BUFFER EMPTY	• TRACE or TRACE_SEARCH command Trace buffer is empty. Check trace conditions and execution state, and re-execute. Then display trace information.
42	GUARDED I/O WRITE	An attempt was made to write data to an emulator internal I/O area. Data cannot be written to.

Table 12.1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
43	CAN NOT RECOVER A = xxxxxxxx	An instruction at the address where a breakpoint is specified with the BREAK or BREAK_SEQUENCE command cannot be recovered after GO command execution. As a result, 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	Writing to ROM was attempted or there was a memory error during verification. Check memory.
45	NOT FOUND	The specified data or information was not found. Correctly specify data.
46	BREAK POINT ADDRESS	The contents of the specified address cannot be modified in parallel mode because the memory address is used by the BREAK or BREAK_SEQUENCE command.
47	NOT FTP CONNECTION	The command cannot be executed because the FTP is not connected. Connect the FTP with the FTP command.
48	FTP CONNECTION ALREADY	The FTP has already been connected. Disconnect the FTP and re-enter the command.
50	DMA EXECUTING	The command cannot be executed during DMA operation. Re-enter it after the DMA operation has been completed.
51	DMA GUARDED OR WRITE PROTECT	A guarded area or write-protected area was accessed during the DMA cycle. Check the user program including the DMA cycle.
52	INTERNAL AREA	An attempt was made to access an area other than CS0 to CS3. This area cannot be accessed with this command. Check the specified address.
53	LABEL TABLE OVERFLOW	The assembler label table overflowed. Reduce the declaration and reference points.
54	INVALID CONFIGURATION FILE	The configuration file contains invalid data. Initiate the emulator with a correct configuration file.
55	CONFIGURATION FILE NOT FOUND	Configuration file is not found on the emulator system disk. Initiate the emulator with a system disk containing the configuration file.
56	CONFIGURATION CHECK ERROR	A warm start cannot be performed because the configuration file to be loaded differs from that has been loaded to the emulator. The warm start must be performed for a file which has been loaded to the emulator. Check the file.

Table 12.1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
57	ILLEGAL INSTRUCTION ADDRESS	The contents of the memory address specified with the BREAK or BREAK_SEQUENCE command is a break instruction (H'0000). A breakpoint cannot be specified at this address.
59	TOO MANY CHARACTERS	Too may characters were specified. Check the number of characters.
60	ALL BREAK POINTS DELETED	All software breakpoints specified by the BREAK or BREAK_SEQUENCE command are cancelled.
61	CAN NOT GET INTO PARALLEL MODE	The option which prevents the emulator from entering parallel mode is specified with the GO command. Change the execution mode.
62	LAN BOARD DISCONNECT	This command cannot be executed because the LAN board is not installed. Install the optional LAN board and re-enter the command.
67	LAN I/O ERROR	An attempt was made to access the MCU internal I/O area. This area cannot be accessed with this command. Check the address.
68	INVALID HOST NAME	The specified host name is not defined with the LAN_HOST command. Define the host name with the LAN_HOST command.
69	MAPPING BOUND MUST 128MB	A memory attribute cannot be specified because the memory addresses specified by MAP or MOVE_TO_RAM includes the cache area, reserved area, or through area. Check the specified address.
70	MAPPING BOUND MUST 128kB	Memory attribute was set in 128-kbyte units with the MAP or MOVE_TO_RAM command. For details, refer to the MAP command.
73	BREAK POINT IS DELETED A = xxxxxxxx	A software breakpoint specified at the displayed address was cancelled because the contents of the address was modified. (Warning message)
74	CAN NOT SET A = xxxxxxxx	A breakpoint cannot be specified at the displayed address by the BREAK or BREAK_SEQUENCE command. A hardware error may have occurred or the contents of the memory address may be a break instruction (H'0000). Correct the error, and reload and re-execute the program. (Warning message)
77	ALL BREAK POINTS DELETED	All software breakpoints were cancelled. (Warning message)

Table 12.1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
78	EMULATOR POD BUSY	The emulator pod was processing a break processing in parallel mode, so another command could not be executed. Re-enter the command. This error occurs when breakpoints are set with the BREAK (with number of times) or BREAK_SEQUENCE command.
81	TRACE CONDITION RESET	Satisfied conditions are all reset when parallel mode is entered. When parallel mode is terminated, the conditions are rechecked from the beginning.
82	ODD ADDRESS	An instruction was written to an odd address was written to by the assembler. However, processing is initiated from the odd address.
83	INVALID OPERAND SIZE	The specified operand size is invalid. Processing is performed with the correct size.
84	INVALID ABSOLUTE ADDRESS	An invalid address was specified by the assembler. Processing is performed with the maximum address allowed.
86	INTERNAL AREA	An area other than CS0 to CS3 areas was accessed. Processing specified with the MEMORY command continues normally, but other processing continues for only the CS0 to CS3 areas.

Table 12.2 Host I/O Error Codes

Error Code	Description and Solution
D1	Parity error: The parity bit specified with the HOST command must match the host system specifications.
D2	Overrun error: E7000 control method is not recognized by the host system. Refer to the description of control methods in section 8.4.1, Control Methods.
D3	Framing error: The baud rate and stop bit specified with the HOST command must match the host system specifications.
D4	Load module format error: The load module format of the transferred data is incorrect. Check the data contents.
DC	Timeout error: Check the connection between the E7000 and host system. Also check the operational status of the host system.
B0	System load error: The system program file does not exist or the environment variable for the IBM PC interface software is not set correctly. Check if the system program is installed correctly.

Table 12.3 Floppy Disk I/O Error Codes

Error Code	Description and Solution
01	A non-existent command was issued to the floppy disk controller (FDC). Reload and re-initiate the E7000 program.
02	A non-existent disk drive is specified. Reload and re-initiate the E7000 program.
03	An invalid sector number was accessed. Reload and re-initiate the E7000 program.
05	The command next was issued during FDC command execution. Reset the E7000 by switching it off and on.
07	File attribute is READ ONLY: Data cannot be written to this file. Remove the write protection or change floppy disks.
0A	Floppy disk directory area is full. Use a new disk.
11	No floppy disk: Insert a floppy disk.
12	Floppy disk is write-protected. Remove the write protection.
21	Data transfer between a floppy disk and memory failed. Retry.
32	The sector to be accessed was not found. Retry.
33	Deleted Data Mark was detected. Reformat the floppy disk.
41	Seek error: Retry.
51	The floppy disk remains in the busy state. Reset the E7000 by switching it off and on.
52	A FAULT signal was set from the FDC. Reset the E7000 by switching it off and on.
53	End of sector was detected. Reset the E7000 by switching it off and on.
54	FDC error: Reset the E7000 by switching it off and on.
55	FDC operation was requested again during FDC operation. Reset the E7000 by switching it off and on.
56	DMAC error: Reset the E7000 by switching it off and on.
C1	Record is too long and cannot be accessed. Check data contents.
C2	End of File was detected. The specified cluster number is incorrect. Correctly specify the cluster number.
C3	End of Volume was detected. The specified sector number is incorrect. Correctly specify the sector number.
CD	The floppy disk is full. Replace with a new floppy disk.

Table 12.4 Floppy Disk Error Messages

Message	Description and Solution
*** FD NOT READY	No floppy disk: Insert a floppy disk and retry.
*** FD NOT SYSTEM FD	The inserted floppy disk is not an E7000 system disk. Insert an E7000 system disk and retry.
*** FD WRITE PROTECT	Data cannot be written because the floppy disk is write-protected. Remove the write protection.
*** FD FORMAT TYPE ERROR	The inserted floppy disk is not compatible with the E7000 or is not formatted. Insert an E7000 disk or reformat the disk.
*** FD CRC ERROR	A CRC error occurred during read/write to the floppy disk. Reformat or replace the disk.
*** FD I/O ERROR nn	An error occurred during read/write to the floppy disk. Error code nn gives details of the error. See table 12.3.

The E7000 system program outputs LAN I/O error messages in the format below. Table 12.5 lists the error messages with brief descriptions.

```
LAN I/O ERROR (E0xx)
socket library error n : <error message>
```

xx: Process in which error occurred (see table 12.6)
n: Error code
<error message>: Refer to table 12.5

If an error message other than that listed in table 12.5 is displayed, refer to the description for the host system error messages.

Table 12.5 LAN I/O Error Messages

Error No.	Error Message	Description
01	not listen	The socket cannot be created
02	In sufficient 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 failure	A connection failure has occurred
09	Illegal IP Address	An illegal IP address has been specified

Table 12.5 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 has already been defined
13	HOSTS Name Nothing	The host 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	An error has occurred during LAN board initialization
20	Illegal 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 be been defined
34	transmission error	A data transfer error has occurred
35	closing error	A termination error has occurred
FF	unknow error	An undefined error has occurred

Table 12.6 Process Code for LAN I/O Error Message

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	Termination

12.2 IBM PC Interface Software Error Messages

The IBM PC interface software outputs error messages on the IBM PC. Table 12.7 lists error messages, descriptions of the errors, and error solutions.

Table 12.7 Interface Software Error Messages

Error Message	Description and Solution
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-SYNTAX ERROR	An error exists in the IBM PC file name. Refer to the debugger and IBM PC manuals and specify with a correct file name.
INTFC ERROR-FILE NOT FOUND	The specified IBM PC file cannot be found or an error is detected in the file name during load.
INTFC ERROR-FILE OPEN ERROR	The directory to which the specified IBM PC file is to be saved is full or an erroneous file name is specified.
INTFC ERROR-FILE READ ERROR	An error has occurred while reading an IBM PC file.
INTFC ERROR-FILE WRITE ERROR	An error has occurred while writing an IBM PC file. Available memory on the disk is insufficient.
INTFC ERROR-FILE CLOSE ERROR	An error has occurred while closing an IBM PC file.
INTFC ERROR-TIMEOUT ERROR	A timeout error has occurred during file transfer or data transfer from the debugger. Check the cable connection and re-transfer.
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.
INTFC ERROR-ABORT BY BREAK	The file transfer has been forcibly terminated by pressing the (BREAK), (STOP), or (CTRL) + C keys.
INTFC ERROR-INVALID COMMAND	An invalid command has been received from the debugger.
INTFC ERROR-EMULATOR NOT READY	The debugger power has been turned off or a cable connected to the debugger has been disconnected. Check that 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-FILE RENAME ERROR	An error has occurred while changing an IBM PC file name.
INTFC ERROR-FILE DELETE ERROR	An error has occurred while deleting an IBM file.

Table 12.7 Interface Software Error Messages (cont)

Error Message	Description and Solution
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-ALREADY ASSIGNED	The specified command is already being executed. Re-execute the command after command execution has been completed.
INTFC ERROR-ENVIRONMENT NOT SPECIFIED	The specified environment variable name could not be detected. Specify the environment variable name with the SET command.
INTFC ERROR-NO INTERFACE BOARD	The interface board is not installed in the IBMPC expansion slot. Check the DIP switch setting on the interface board and that the interface board is inserted in the expansion slot correctly, and retransfer. If the same error occurs again, inform a Hitachi sales agency.

Appendix A Emulator External Dimensions and Weight

Figures A.1 and A.2 show the emulator external dimensions and weight.

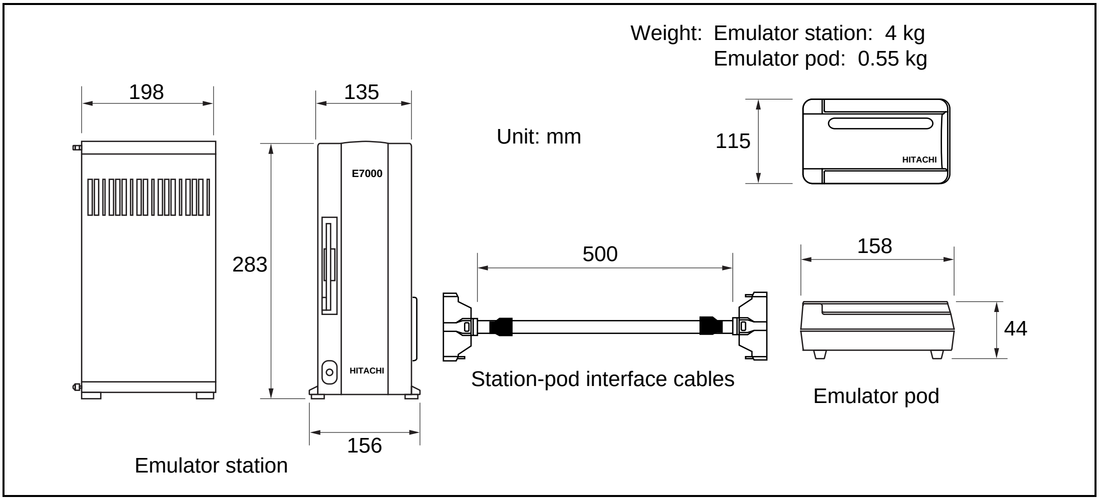


Figure A.1 E7000 External Dimensions and Weight

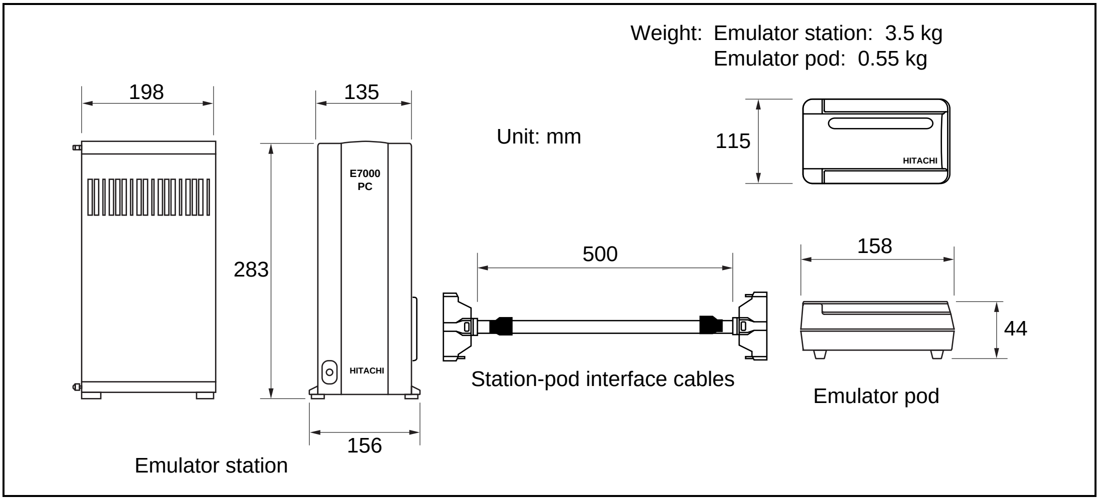


Figure A.2 E7000PC External Dimensions and Weight

Table A.1 shows the emulator user interface pin alignment.

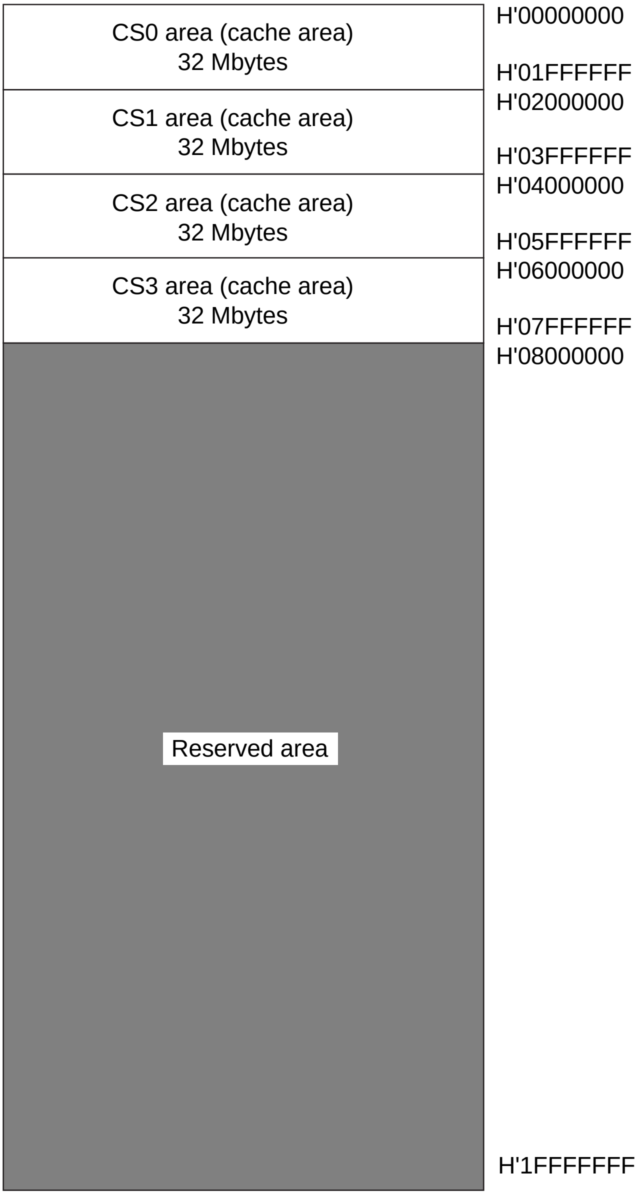
Table A.1 SH-2 Emulator User Interface Pin Alignment

PGA 171	QFP 144	Signal Name	PGA 171	QFP 144	Signal Name	PGA 171	QFP 144	Signal Name	PGA 171	QFP 144	Signal Name
1		VSS	44		VSS	87		VSS	130		VSS
2	1	D11	45	37	A6	88	73	CS1	131	110	MD2
3	2	D12	46	38	A7	89	74	CS2	132	111	CKPACK
4	3	D13	47	39	A8	90	75	CS3	133	112	CKPREQ
5	4	VCC	48	40	VCC	91	76	BS	134	113	VCC
6	5	D14	49	41	A9	92	77	RDWR	135	114	EXTAL
7	6	VSS	50	42	VSS	93	78	VSS	136	115	VSS
8		VSS	51		VSS	94		VSS	137	116	XTAL
9	7	D15	52	43	A10	95	79	RAS	138		VSS
10	8	D16	53	44	A11	96	80	CAS	139	117	MD3
11	9	D17	54	45	A12	97	81	WE3	140	118	CKIO
12	10	D18	55	46	A13	98	82	WE2	141	119	MD4
13	11	D19	56	47	A14	99	83	WE1	142	120	MD5
14	13	D20	57	49	A15	100	85	WE0	143	121	VSS
15	14	VSS	58	50	VSS	101		VSS	144	122	RESET
16	12	VCC	59	48	VCC	102	84	VCC	145		VSS
17		VSS	60		VSS	103	86	VSS	146	123	VCC
18	15	D21	61	51	A16	104	87	RD	147	124	IVECF
19	16	D22	62	52	A17	105	94	WDTOVF	148	125	NMI
20	17	D23	63	53	A18	106	89	WAIT	149	126	IRL3
21	18	VCC	64	54	VCC	107	90	BEN	150		VSS
22		VSS	65		VSS	108	91	VSS	151	127	IRL2
23	20	VSS	66	56	VSS	109		VSS	152	128	IRL1
24	19	D24	67	55	A19	110	92	BACK	153	129	IRL0
25	21	D25	68	57	A20	111	93	BREQ	154		VCC
26	22	D26	69	58	A21	112	96	VCC	155		VSS
27	23	D27	70	59	A22	113		VSS	156	130	D0
28	24	VCC	71	60	VCC	114	88	CKE	157		VSS
29		VSS	72		VSS	115	98	VSS	158	131	D1
30	25	D28	73	61	A23	116	95	FTOB	159	133	D2
31	26	VSS	74	62	VSS	117	97	FTOA	160	135	D3
32	27	D29	75	63	A24	118	99	FTI	161	136	D4
33	28	D30	76	64	A25	119	100	FTCI	162	137	D5
34	29	D31	77	65	A26	120	101	RXD0	163	138	D6
35	30	A0	78	66	DACK0	121	102	TXD0	164		VSS
36		VSS	79		VSS	122		VSS	165	139	VCC
37	31	A1	80	67	VCC	123	103	SCK0	166	140	D7
38	32	A2	81	68	DACK1	124	104	VCC	167		VSS
39	33	VSS	82	69	VSS	125	106	VSS	168	142	D8
40	34	A3	83	70	DREQ0	126	105	MD0	169	143	D9
41	35	A4	84	71	DREQ1	127	107	MD1	170	144	D10
42	36	A5	85	72	CS0	128		VSS	171		VSS
43		VSS	86		VSS	129		VSS			

Note: +5 V power supply must be applied to pin 124 (Vcc).

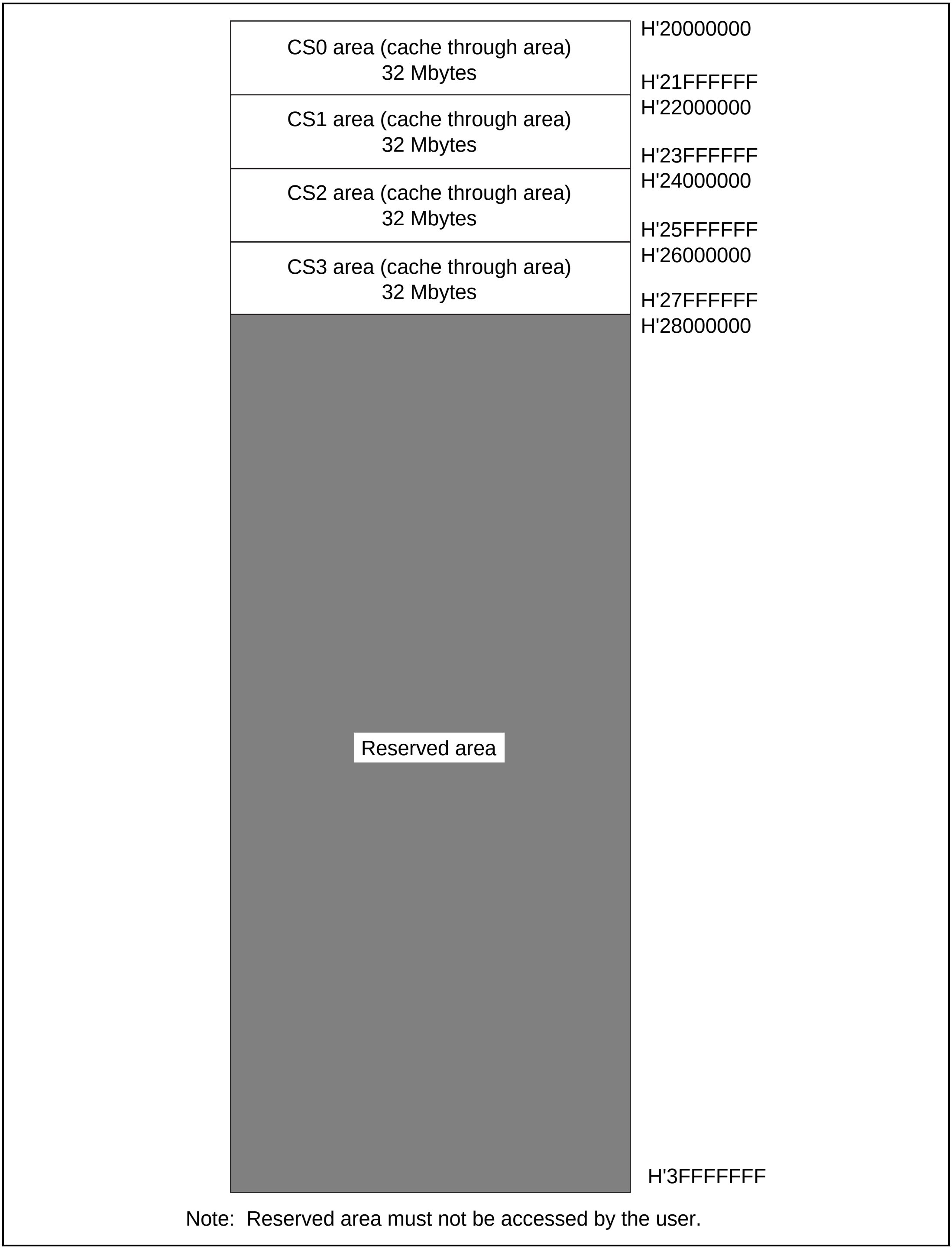
Appendix B Memory Map

This section shows memory map in each area.



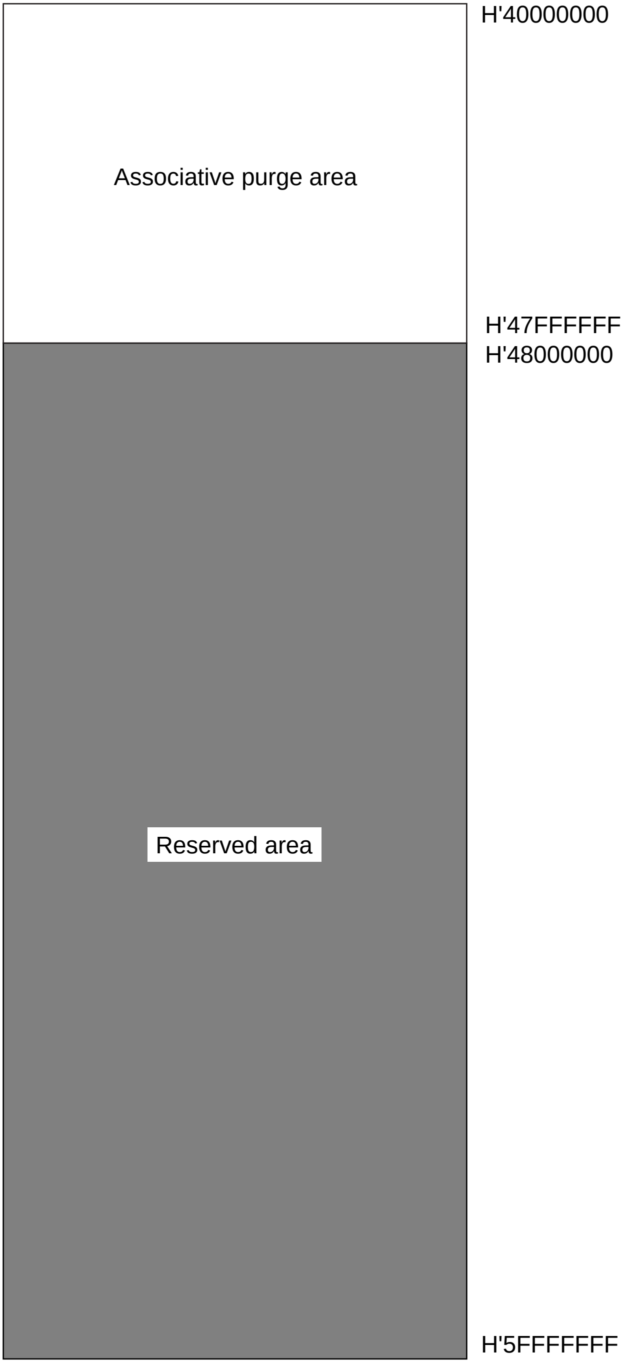
Note: Reserved area must not be accessed by the user.

Figure B.1 Cache Area Memory Map



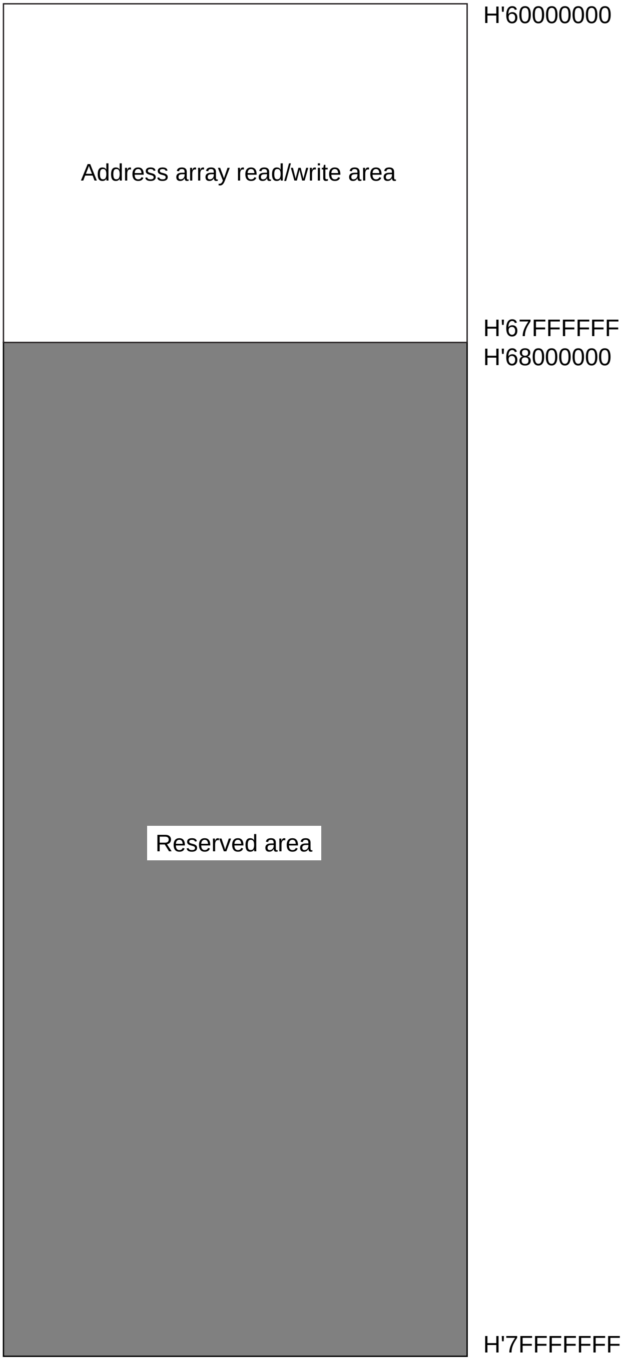
Note: Reserved area must not be accessed by the user.

Figure B.2 Memory Map for Cache Through Area



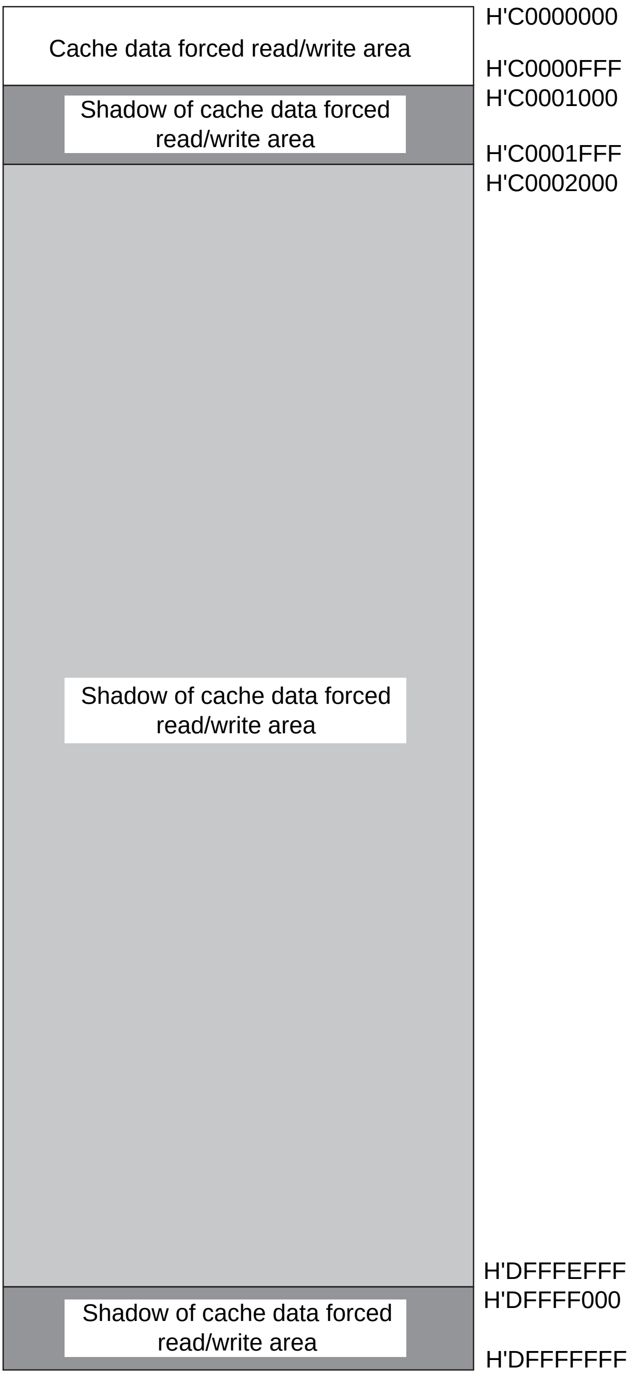
Note: Reserved area must not be accessed by the user.

Figure B.3 Memory Map for Associative Purge Area



Note: Reserved area must not be accessed by the user.

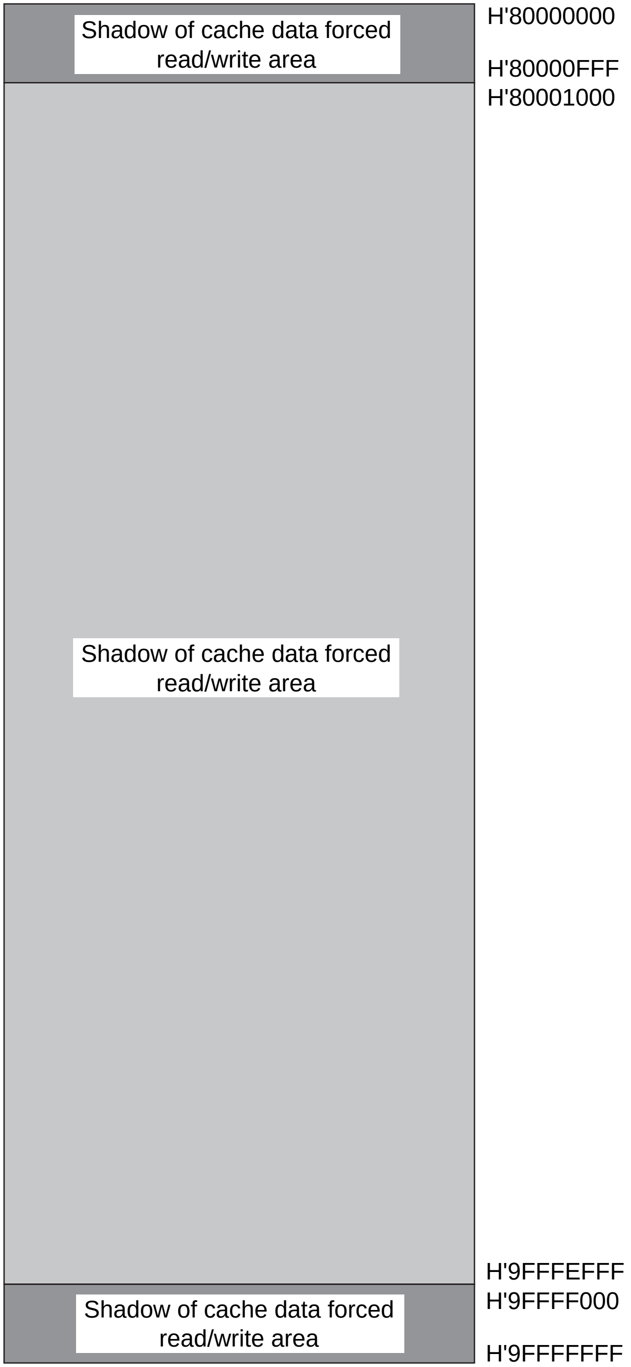
Figure B.4 Memory Map for Address Array Read/Write Area



H'C0000000
Cache data forced read/write area
H'C0000FFF
H'C0001000
Shadow of cache data forced read/write area
H'C0001FFF
H'C0002000
Shadow of cache data forced read/write area
H'DFFFEFFF
H'DFFFF000
Shadow of cache data forced read/write area
H'DFFFFFFF

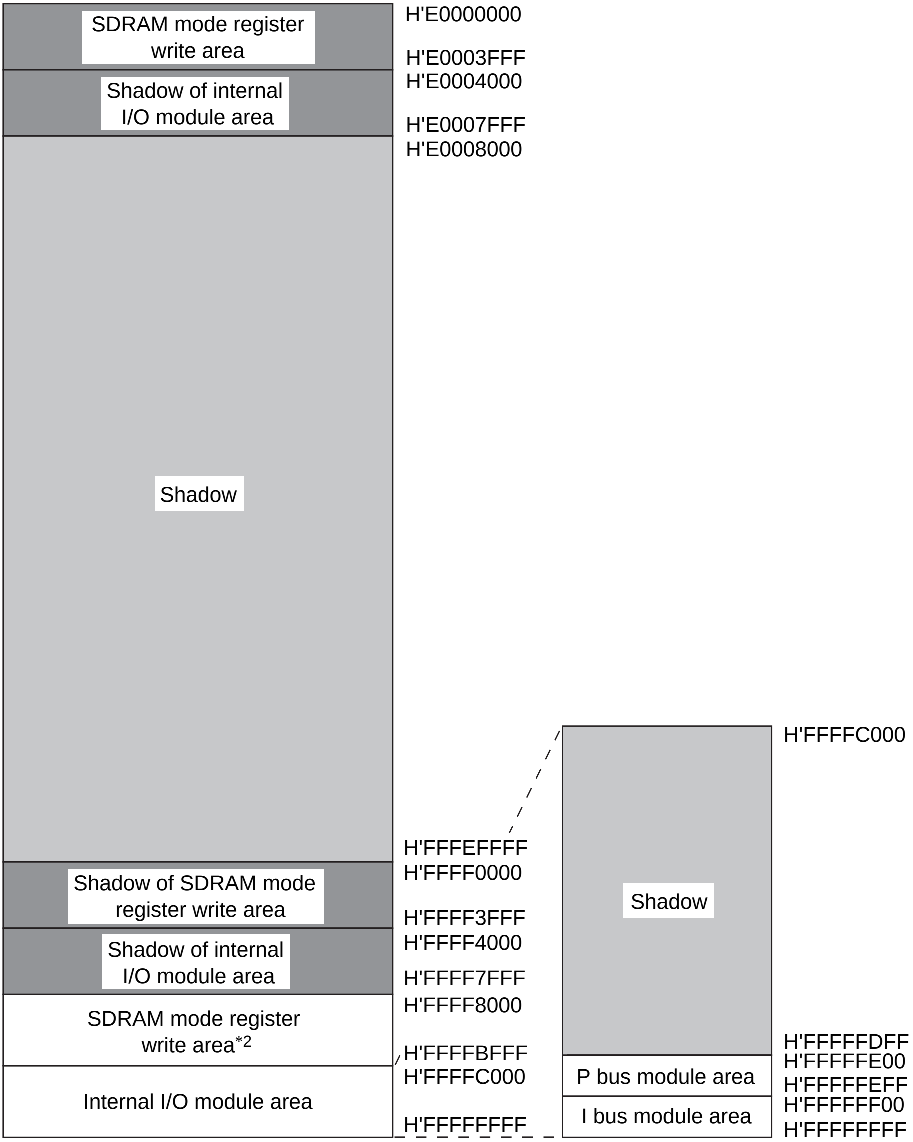
Note: Shadows are reserved areas (must not be accessed by the user).

Figure B.5 Memory Map for Cache Data Forced Read/Write Area


H'80000000
Shadow of cache data forced read/write area
H'80000FFF
H'80001000
Shadow of cache data forced read/write area
H'9FFFEFFF
H'9FFFF000
Shadow of cache data forced read/write area
H'9FFFFFFF

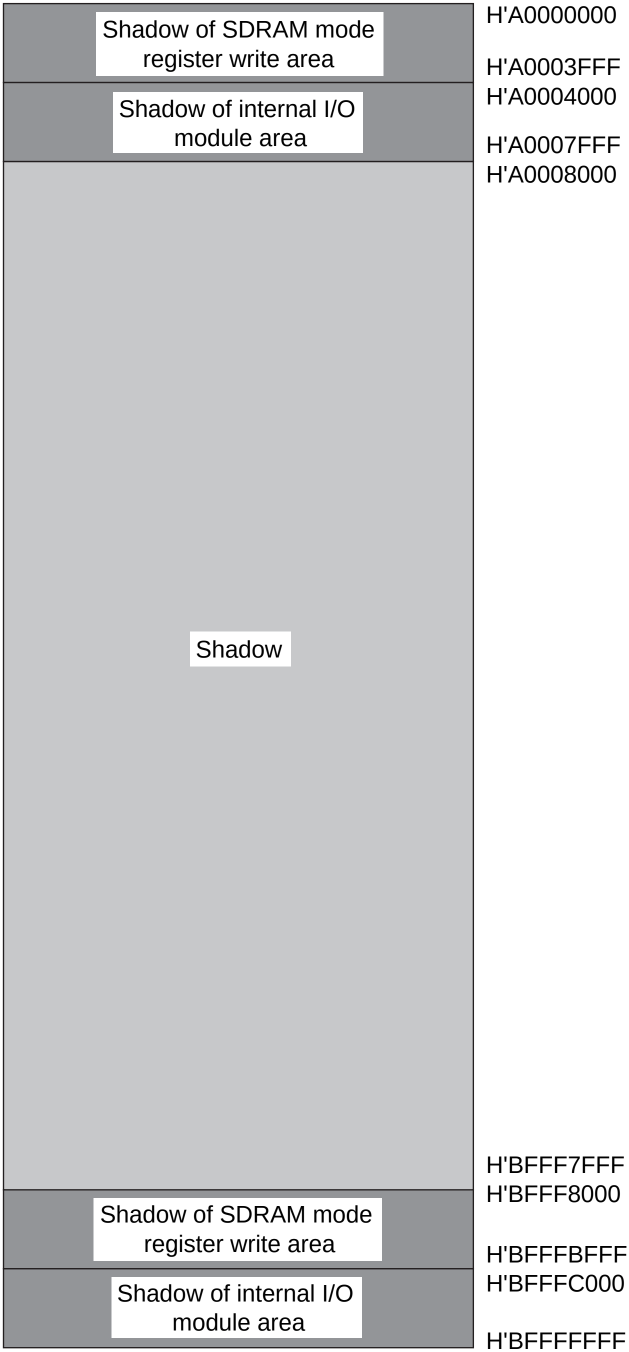
Note: Shadows are reserved areas (must not be accessed by the user).

Figure B.6 Memory Map for Cache Data Forced Read/Write Area (2)



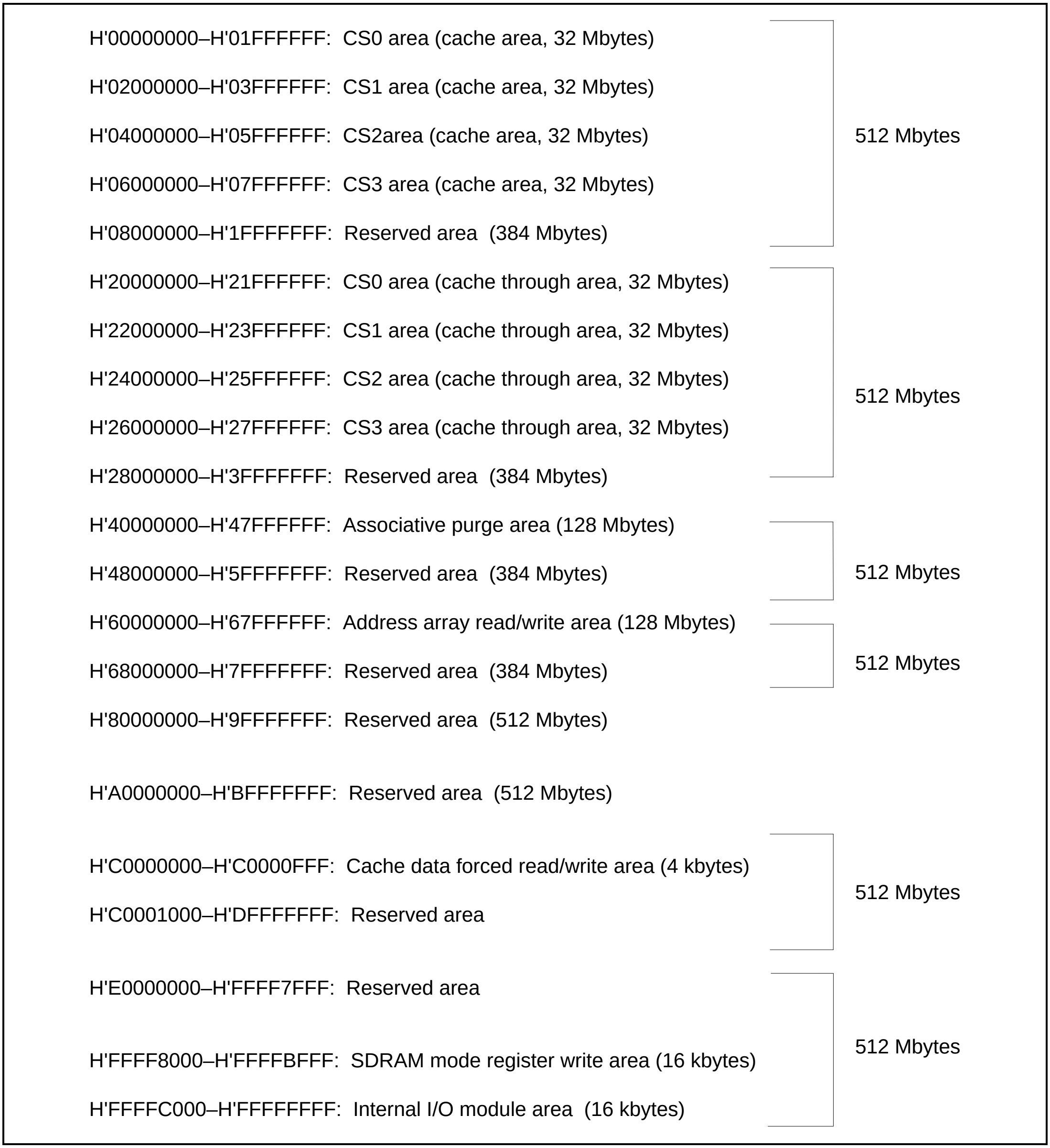
Notes: 1. Shadows are reserved areas (must not be accessed by the user).
2. SDRAM mode register write area cannot be read.

Figure B.7 Memory Map for Internal I/O Module Area


Shadow of SDRAM mode register write area
Shadow of internal I/O module area
Shadow
Shadow of SDRAM mode register write area
Shadow of internal I/O module area
H'A0000000
H'A0003FFF
H'A0004000
H'A0007FFF
H'A0008000
H'BFFF7FFF
H'BFFF8000
H'BFFFBFFF
H'BFFFC000
H'BFFFFFFF

Note: Shadows are reserved areas (must not be accessed by the user).

Figure B.8 Memory Map for Internal I/O Module Area (2)



H'00000000–H'01FFFFFF: CS0 area (cache area, 32 Mbytes)

H'02000000–H'03FFFFFF: CS1 area (cache area, 32 Mbytes)

H'04000000–H'05FFFFFF: CS2area (cache area, 32 Mbytes)

H'06000000–H'07FFFFFF: CS3 area (cache area, 32 Mbytes)

H'08000000–H'1FFFFFFF: Reserved area (384 Mbytes)

512 Mbytes

H'20000000–H'21FFFFFF: CS0 area (cache through area, 32 Mbytes)

H'22000000–H'23FFFFFF: CS1 area (cache through area, 32 Mbytes)

H'24000000–H'25FFFFFF: CS2 area (cache through area, 32 Mbytes)

H'26000000–H'27FFFFFF: CS3 area (cache through area, 32 Mbytes)

H'28000000–H'3FFFFFFF: Reserved area (384 Mbytes)

512 Mbytes

H'40000000–H'47FFFFFF: Associative purge area (128 Mbytes)

H'48000000–H'5FFFFFFF: Reserved area (384 Mbytes)

512 Mbytes

H'60000000–H'67FFFFFF: Address array read/write area (128 Mbytes)

H'68000000–H'7FFFFFFF: Reserved area (384 Mbytes)

512 Mbytes

H'80000000–H'9FFFFFFF: Reserved area (512 Mbytes)

H'A0000000–H'BFFFFFFF: Reserved area (512 Mbytes)

H'C0000000–H'C0000FFF: Cache data forced read/write area (4 kbytes)

H'C0001000–H'DFFFFFFF: Reserved area

512 Mbytes

H'E0000000–H'FFFF7FFF: Reserved area

H'FFFF8000–H'FFFFBFFF: SDRAM mode register write area (16 kbytes)

H'FFFFC000–H'FFFFFFFF: Internal I/O module area (16 kbytes)

512 Mbytes

Figure B.9   Memory Map

Appendix C ASCII Codes

Upper 4 bits Lower 4 bits	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	'	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	’	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	–	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

E7000 SH7604 Emulator User's Manual

Publication Date: 1st Edition, June1994
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.