Hitachi Microcomputer Support Hardware

E7000 H8S/2655 Series Emulator

User’s Manual

HITACHI HS2655EPD70HE

IMPORTANT INFORMATION

READ FIRST

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

- **KEEP the user's manual handy for future reference.**

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

Emulator Product:

Throughout this document, the term "emulator product" shall be defined as the emulator station, emulator pod, and cables, produced only by Hitachi, Ltd excluding all subsidiary products.
The user system or a host computer is not included in this definition.

Purpose of the Emulator Product:

This emulator product is a software and hardware development tool for systems employing the Hitachi microcomputer H8S/2655 series and H8S/2245 series (hereafter referred to as MCU). By exchanging the emulator pod, this emulator product can also be used for systems using other E7000-series microcomputers. This emulator product must only be used for the above purpose.

Limited Applications:

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

Improvement Policy:

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

Target User of the Emulator Product:

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

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

LIMITED WARRANTY

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

DISCLAIMER

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

State Law:

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

The Warranty is Void in the Following Cases:

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

All Rights Reserved:

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

Other Important Things to Keep in Mind:

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

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

Figures:

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

Limited Anticipation of Danger:

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

SAFETY PAGE

READ FIRST

- **READ this user's manual before using this emulator product.**
- **KEEP the user's manual handy for future reference.**

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

DEFINITION OF SIGNAL WORDS

DANGER indicates an **imminently** hazardous situation which, **if not avoided**, will result in **DEATH** or **SERIOUS INJURY** to you or other people.

WARNING indicates a **potentially** hazardous situation which, **if not avoided**, could result in **DEATH** or **SERIOUS INJURY** to you or other people.

CAUTION indicates a hazardous situation which, **if not avoided**, may result in **minor or moderate injury** to you or other people, or may result in **damage to the machine** or **loss of the user program**. It may also be used to alert against unsafe usage.

NOTE emphasizes essential information.

⚠ WARNING

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

1. Always switch OFF the emulator product and the user system before connecting or disconnecting the EMULATOR STATION CABLES or the EMULATOR POD.

2. When connecting the emulator pod to the user system, ensure that pin 1 of the user system connector on the emulator pod and that on the user system IC socket are correctly aligned.

Warnings on Emulator Usage

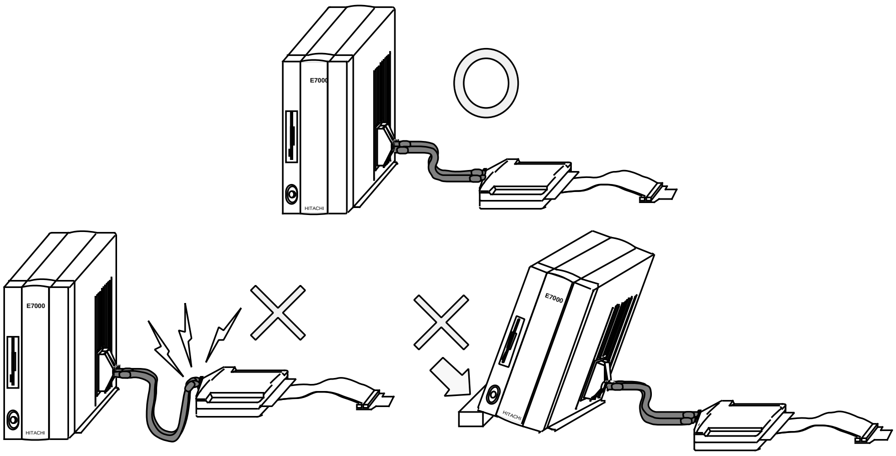
Warnings described below apply as long as you use the E7000 or E7000PC emulator. Be sure to read and understand the warnings below before using these emulators. Note that these are the main warnings, not the complete list.

⚠ WARNING

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

⚠ WARNING

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



Preface

Thank you for purchasing the emulator for the Hitachi's original microcomputer H8S/2655 series and H8S/2245 series.

CAUTION

READ this User's Manual before using the emulator product. To use the E7000, read section 3, Preparation before Use of Part I, E7000 Guide. To use the E7000PC, read section 3, Preparation before Use of Part II, E7000PC Guide. Incorrect operation will damage the user system and the emulator product.

The emulator is an efficient software and hardware development tool for systems based on Hitachi's original microcomputer H8S/2655 series or H8S/2245 series (hereafter collectively referred to as MCU). By exchanging the emulator pod, this emulator can also be used for other H-series microcomputers.

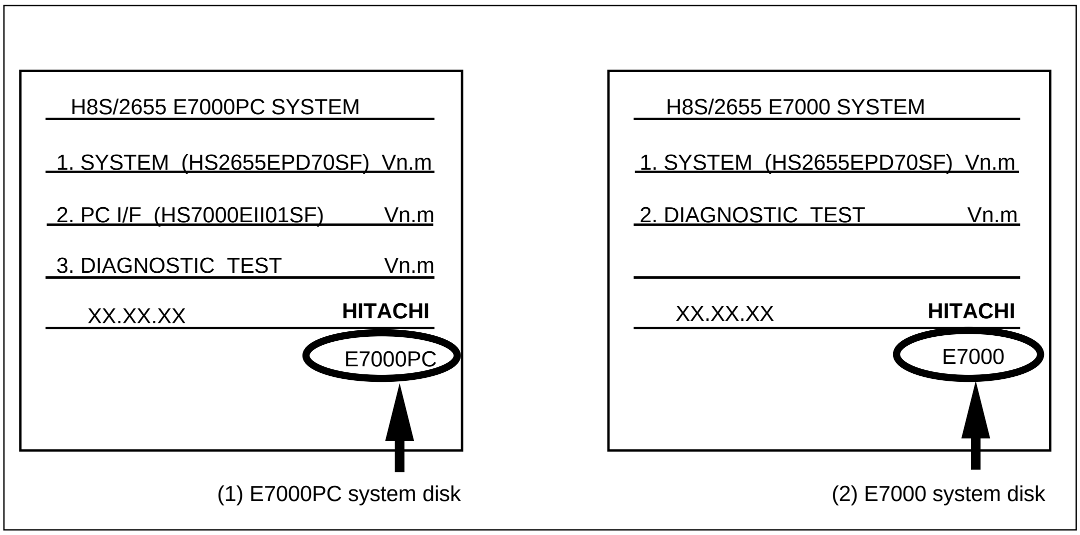
There are two types of H8S/2655 emulators: the E7000 and the E7000PC, which is used only with the IBM PC*. This manual describes the functions and operating procedures of the E7000 and E7000PC for the H8S/2655 series and H8S/2245 series.

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.

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

Two system floppy disks are packaged together with the H8S/2655 emulator pod. Use the floppy disk corresponding to your emulator station, E7000 or E7000PC (the disks are labeled accordingly, as shown in the figure).



(1) E7000PC system disk

(2) E7000 system disk

Figure System Disk Labels

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

When Using 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 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 E7000PC. For details on the copy procedure, refer to section 3.4, System Software Installation in Part II, E7000PC Guide.

Related Manuals:

H8/300-Series Cross Assembler User’s Manual
H8/300-Series C Compiler User’s Manual
H-Series Linkage Editor User’s Manual
H-Series Librarian User’s Manual
H8/300-Series Simulator/Debugger User's Manual
H8S/2655 E7000PC Graphical User Interface Software User’s Manual
Integrated Development Manager User's Manual
HS7000EST01H Manual
HS7000ESTP1H Manual
LAN Board Manual
Description Notes on Using the IBM PC Interface Board (HS7000EII01H)
Memory Board Instruction Manual
Bus Monitor Instruction Manual
Bus Monitor Interface Board Instruction Manual

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

H-Series Interface Software User’s Manual

Contents

Part I E7000 Guide

Part II E7000PC Guide

Part III Emulator Function Guide

Detailed 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 H8S/2655-series or H8S/2245-series microcomputer (collectively referred to as MCU), a member of H8S-series microcomputers, developed by Hitachi, Ltd.

The H8S/2655 series contains the high-speed CPU, internal RAM and ROM, a timer unit, a watchdog timer, serial communication interface, DMAC, DTC, I/O ports, and A/D and D/A converters on a single chip.

The H8S/2245 series contains the high-speed CPU, internal RAM and ROM, a timer unit, a watchdog timer, serial communication interface, DTC, I/O ports, and A/D and D/A converters on a single chip.

When the E7000 is connected to a user system, it operates in place of the 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, emulator pod, and user system interface cable, as shown in figure 1-1.

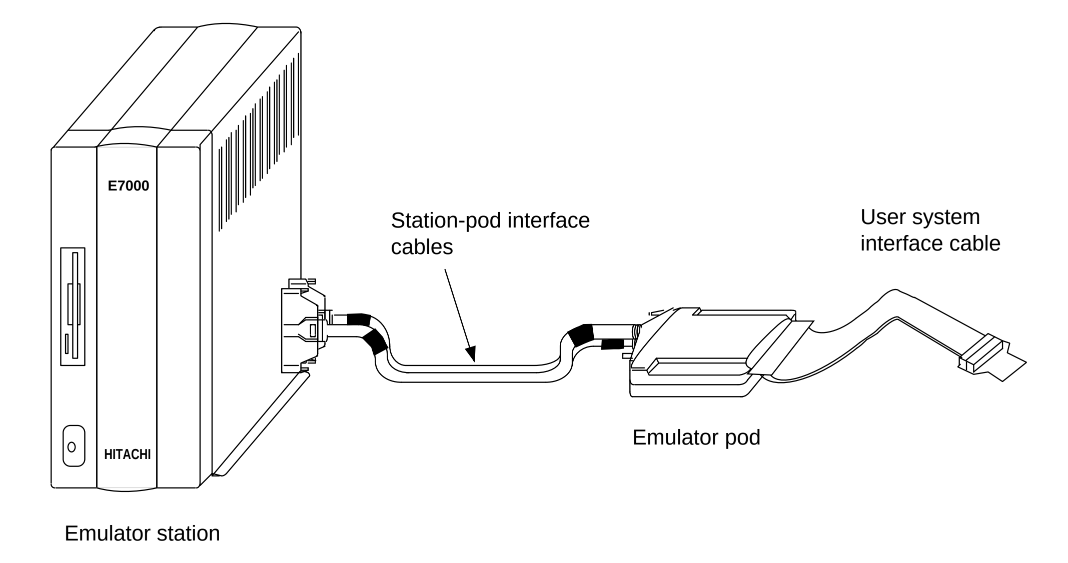


Figure 1-1 H8S/2655 Series E7000 Emulator

The E7000 provides the following features:

- Realtime emulation of the MCU
- 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.
- Integrated development manager (IDM: 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 computer 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
- Emulation at a low voltage (2.7 V to 5.5 V) supplied from the user system.

1.1 Warnings

CAUTION

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

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 at the rear panel.

7. After connecting a 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 MCU and the E7000, refer to section 2, Differences between the Actual MCU and the Emulator in Part III, Emulator Function Guide.

Floppy Disk:

CAUTION

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

1.2 Environmental Conditions

CAUTION

If these instructions are not followed when using the emulator product, the user system and the emulator product will be damaged. The USER PROGRAM will be lost.

Observe the conditions listed in table 1-1 when using the E7000.

Table 1-1 Environmental Conditions

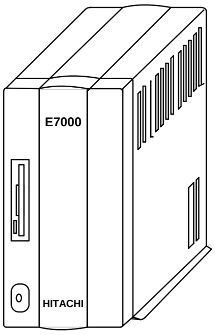
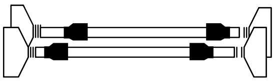
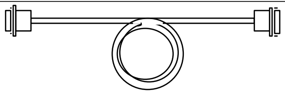
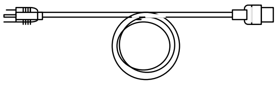
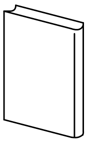
Item	Specifications
Temperature	Operating: +10 to +35°C Storage: –10 to +50°C
Humidity	Operating: 35 to 80% RH no condensation Storage: 35 to 80% RH no condensation
Vibration	Operating: 2.45 m/s^2 max. Storage: 4.9 m/s^2 max. Transportation: 14.7 m/s^2 max.
AC input power	Voltage: 100/200 VAC ±10% Frequency: 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.

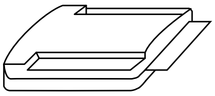
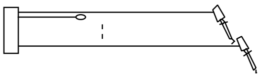
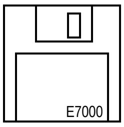
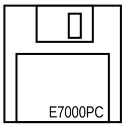
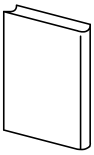
1.3.1 E7000 Emulator Station

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 	1	
	Fuse 	1	Spare (3 A)
Documentation	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
	External probe		1	Signal input: Eight GND: One Trigger output: One
Software	Floppy disks		1	E7000 system program
			1	E7000PC system program* (cannot be used with the E7000)
Documentation	H8S/2655 Series E7000 Emulator User’s Manual		1	HS2655EPD70HE

Note: The E7000PC system program cannot be used with the E7000.

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)
1-Mbyte emulation memory board	HS7000EMS11H	1-Mbyte SRAM is used
4-Mbyte emulation memory board	HS7000EMS12H	4-Mbyte SRAM is used
TQFP-120 user system interface cable	HS2655ECN71H	For H8S/2655 series (TFP-120)
QFP-128 user system interface cable	HS2655ECH71H	For H8S/2655 series (FP-128)
QFP-100 user system interface cable	HS2245ECH71H	For H8S/2245 series (FP-100B and TFP-100B)
Host computer interface cable	HS7000EHT71H	RS-232C interface
Printer cable	HS7000EPR71H	Centronics interface
Bus monitor interface board for E7000	HS7000EXR10H	For connecting the E7000 bus monitor board
E7000 bus monitor board	HS7000EBR01H	For installing the D/A converter

Section 2   Components

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), emulator pod, user system interface cable, and external probes. By installing a LAN board (optional), the emulator station can be connected to a workstation.

To provide additional memory, optional 1-Mbyte and 4-Mbyte SRAM boards are available. The E7000 contains another slot into which you can insert either one of these boards or a bus monitor interface board.

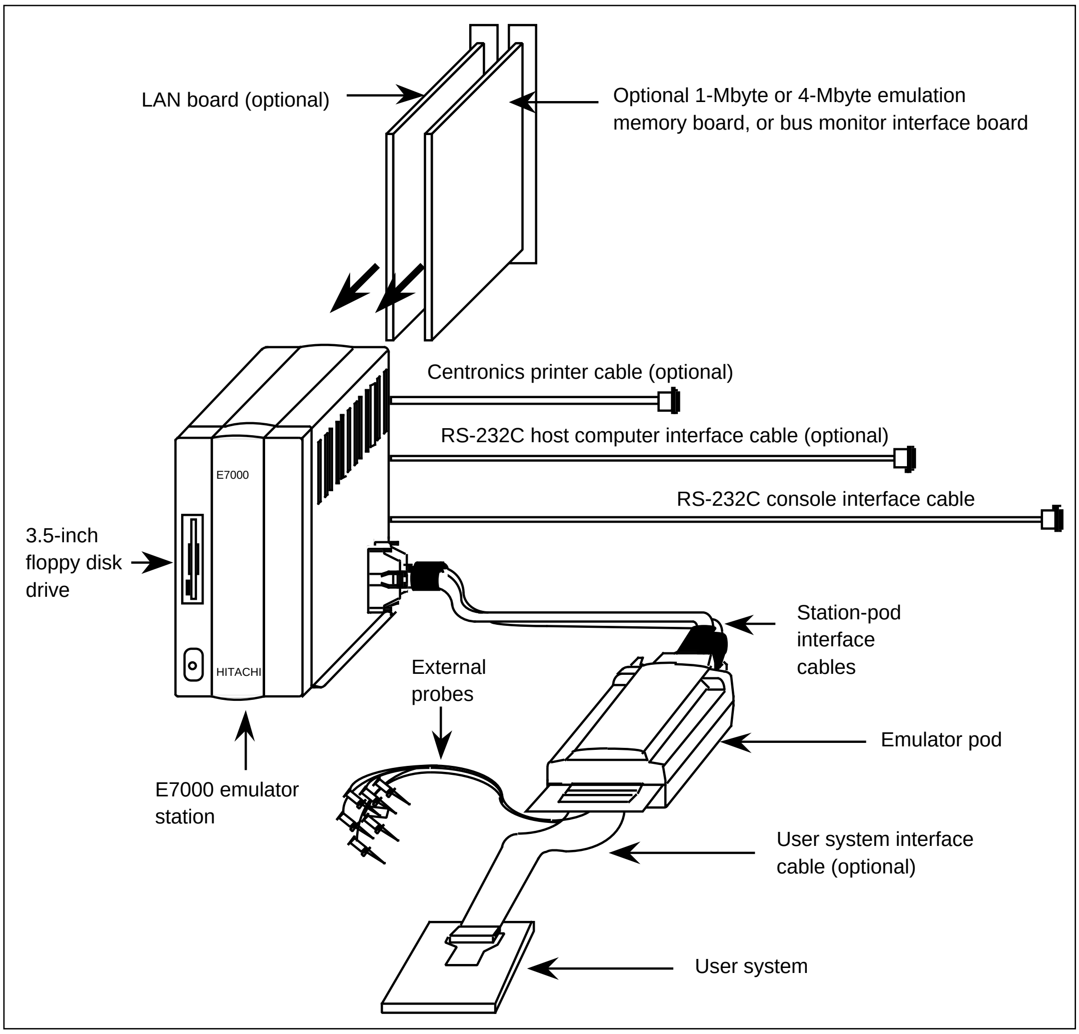

LAN board (optional)
Optional 1-Mbyte or 4-Mbyte emulation memory board, or bus monitor interface board
Centronics printer cable (optional)
RS-232C host computer interface cable (optional)
RS-232C console interface cable
E7000
3.5-inch floppy disk drive
HITACHI
Station-pod interface cables
External probes
Emulator pod
E7000 emulator station
User system interface cable (optional)
User system

Figure 2-1   E7000 Emulator Hardware Components

2.1.1 Emulator Station Components

Front Panel:

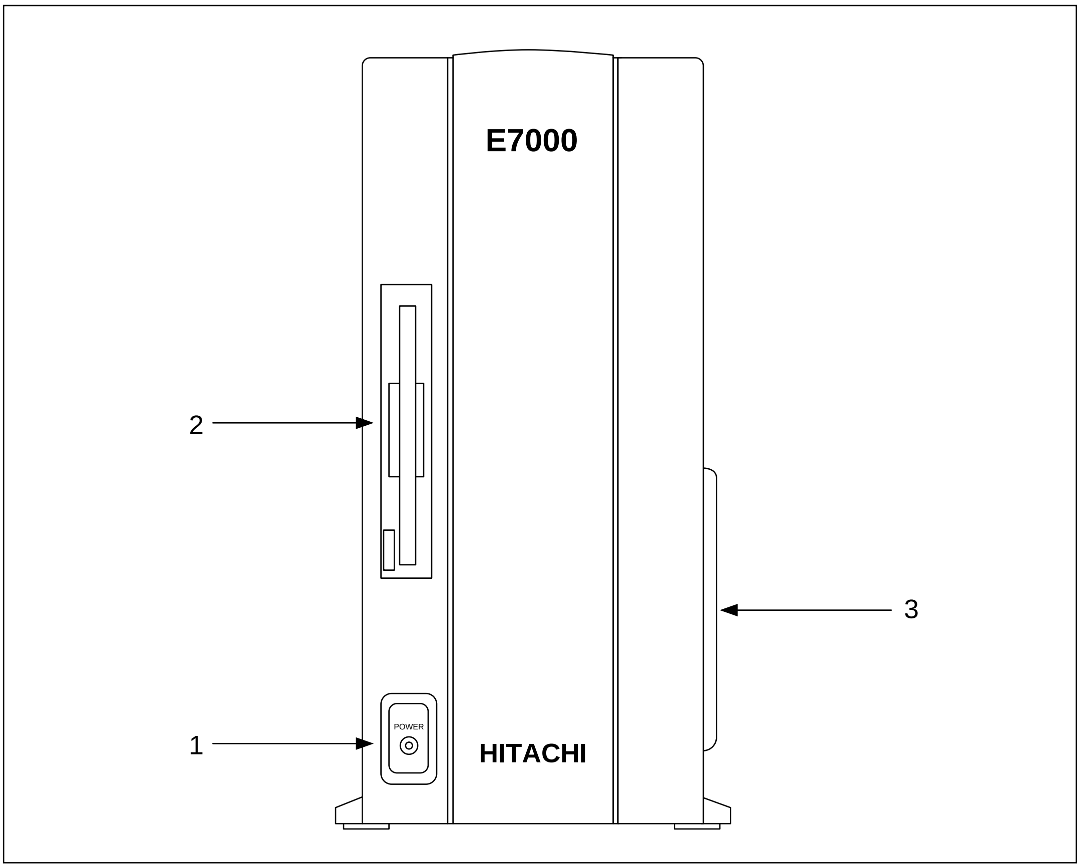


Figure 2-2 E7000 Emulator Station Front Panel

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.

CAUTION

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

3. Station-pod interface cable connectors: For connecting the emulator pod to the emulator station.

⚠ WARNING

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

Rear Panel:

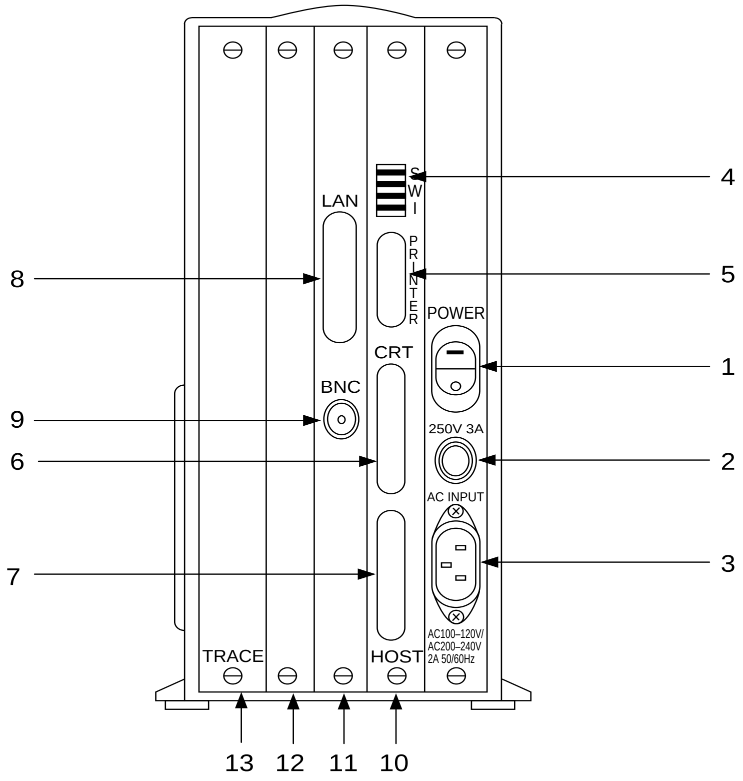


Figure 2-3 E7000 Emulator Station Rear Panel

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

WARNING

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

2. Fuse box: Contains a 3-A 250 VAC fuse.

3. AC power connector: For a 100/200 VAC power supply.

4. Console interface switch: For changing the transfer rate, 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 computer connector: For RS-232C communication with a host computer. 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. Emulation memory/ bus monitor interface board slot: For installing the optional emulation memory board or bus monitor interface board.

13. Trace board slot: For installing the trace board.

2.1.2 Emulator Pod Components

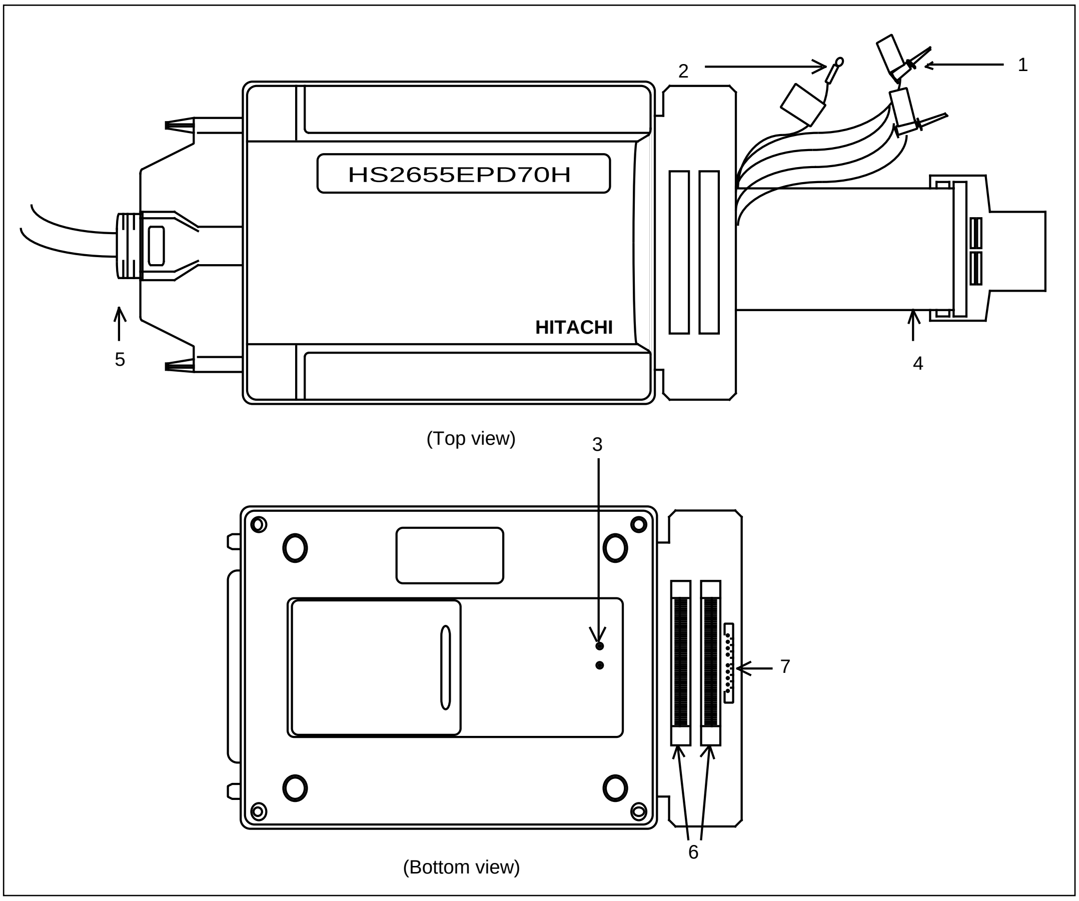


Figure 2-4 E7000 Emulator Pod

1. External probes: Can be used for the following during user system emulation
   - Hardware break condition input
   - Realtime trace input
   - Multibreak detection

2. Trigger output pin: Outputs a low-level pulse in the following states:
   - When a hardware break condition is satisfied (whether or not to break can be selected)
   - When cycle reset mode is specified with the GO command and an RES signal is input to the MCU
   - When trigger output is specified with the TRACE_ CONDITION command and the specified conditions are satisfied, this output can be used as a trigger signal for an oscilloscope or a logic analyzer.

3. Crystal oscillator terminals: For installing a crystal oscillator to be used as a clock source for the MCU.

4. User system interface cable: For connection to the MCU socket on the user system, to enable the E7000 to operate in place of the MCU.

WARNING

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

CAUTION

When a user system interface cable is connected, power supply voltage must be provided from the VCC pin on the user system interface cable head to operate the E7000. Therefore, when operating the E7000 alone, be sure to disconnect the user system interface cable.

5. Station-pod interface cables: For connecting the emulator station to the emulator pod.

WARNING

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

6. User system interface cable connector: For connection of the user system.

⚠ WARNING

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

1. **Always switch OFF the emulator product and the user system before connecting or disconnecting the EMULATOR STATION CABLES or the EMULATOR POD.**

2. **When connecting the emulator pod to the user system, ensure that pin 1 of the user system connector on the emulator pod and that on the user system IC socket are correctly aligned.**

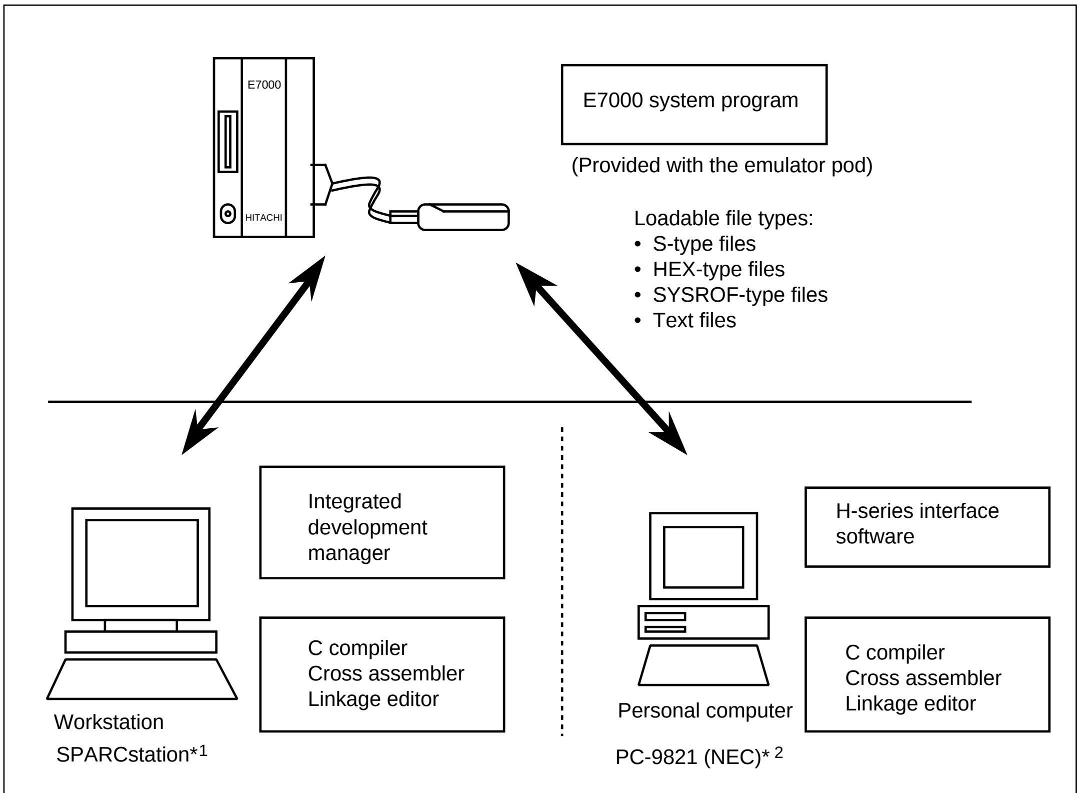
7. External probe connector: For connection of the external probe.

⚠ WARNING

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

2.2 E7000 Software Components

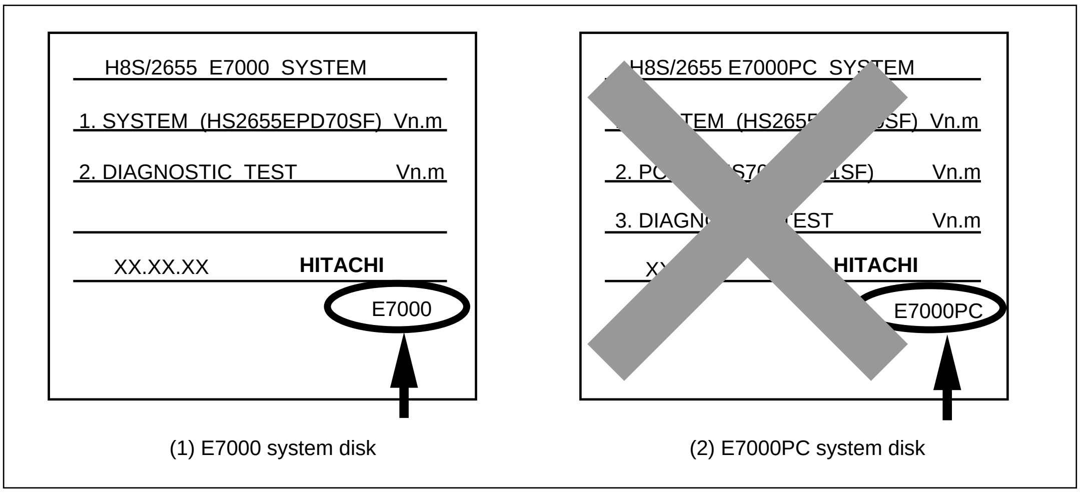
The E7000's software components are illustrated in figure 2-5.



Notes:
1. SPARC is a CPU and workstation administrated by SPARC International, Inc., and is based on the architecture developed by Sun Microsystems, Inc. SPARCstation is a registered trademark of SPARC International, INC. and has permission of monopolistic use granted by Sun Microsystems, Inc.
2. PC-9821 is a registered trademark of NEC Corporation.

Figure 2-5 E7000 Emulator Software Components

The emulator pod contains two 3.5-inch floppy disks: the E7000 emulator system disk has "E7000" written under "HITACHI" on its label. Do not use the disk that has "E7000PC" on its label.



(1) E7000 system disk

(2) E7000PC system disk

Figure 2-6 System Disk Labels

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 emulator station memory after the E7000 system program is activated.
H8S2655P.SYS	MCU control program	Controls the MCU within the emulator pod. Loaded into the emulator station memory after the E7000 system program is activated.
H8S2655C.SYS	Configuration file	Contains MCU operating mode and MAP information. Loaded with the E7000 system program.
LANCNF.SYS	LAN configuration file	Stores the host computer 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.

2.3 System Configuration

The E7000 can be connected with the host computer 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 2-7.

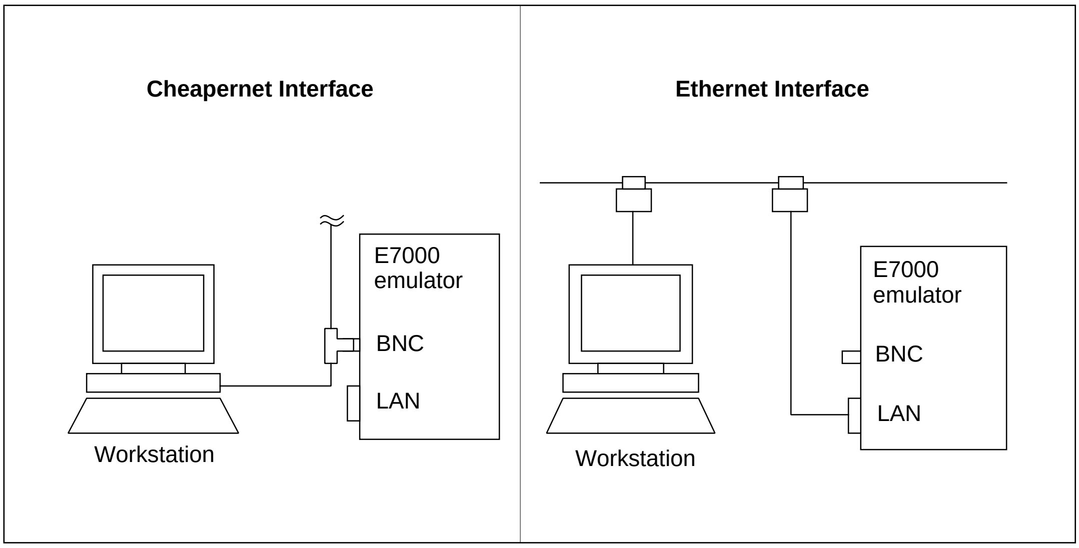


Figure 2-7 System Configuration Using LAN Interface

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 2-8.

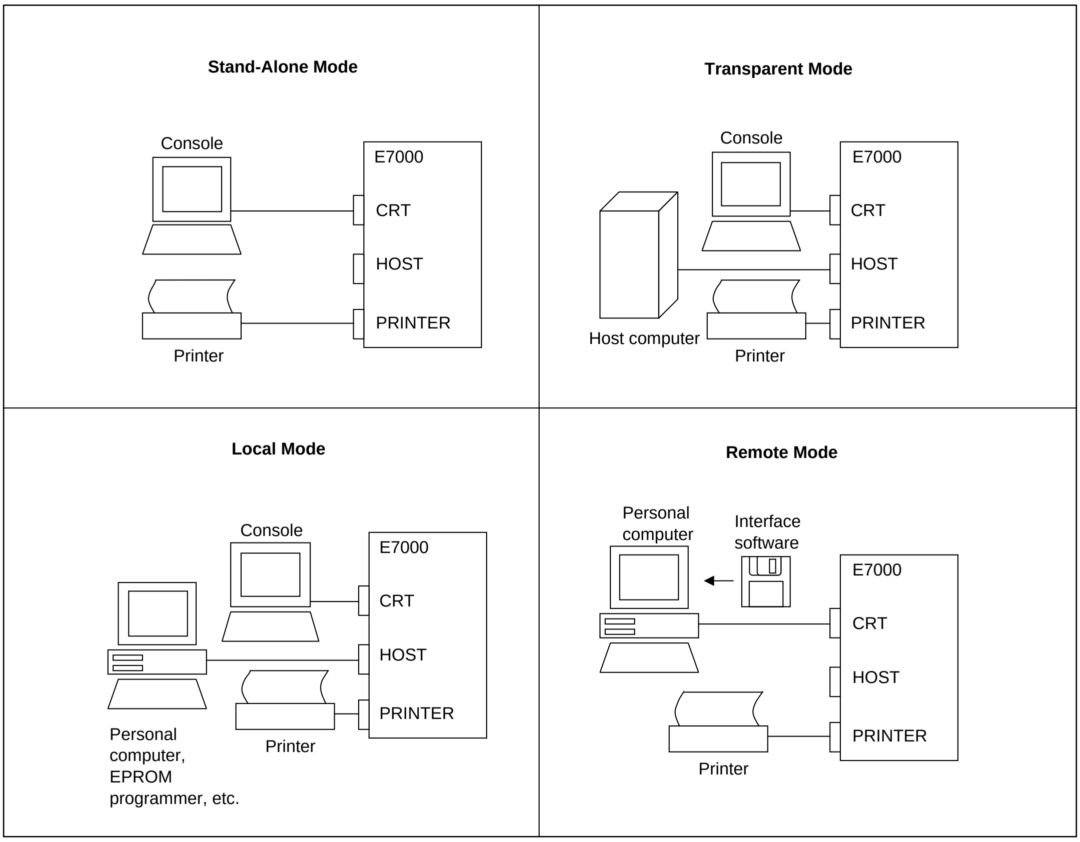


Figure 2-8 System Configuration Using RS-232C Interface

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 computer. 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 computer for data transfer. Interface software must be loaded in the personal computer.

Section 3 Preparation before Use

3.1 E7000 Preparation

CAUTION

READ the reference sections shaded in figure 3-1 and the following warnings before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

Unpack the E7000 and prepare it for use as follows:

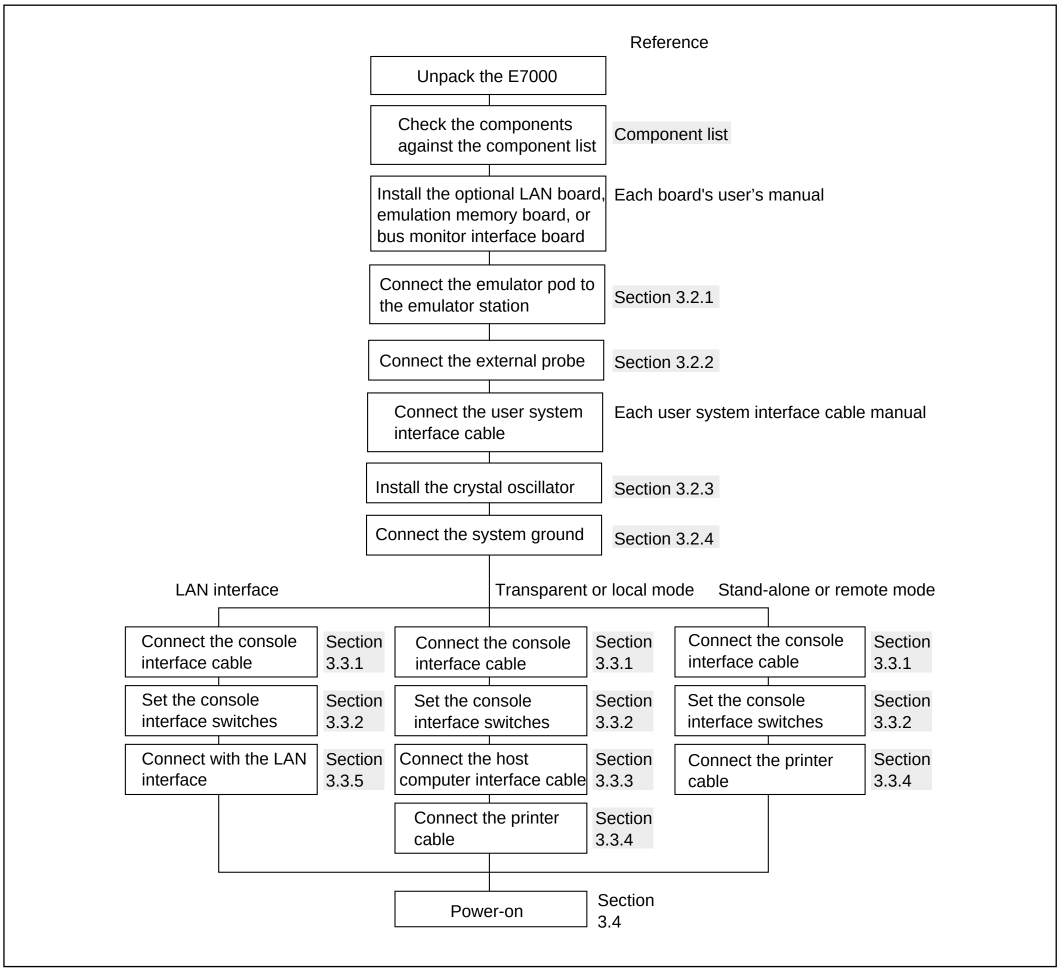

Reference
Unpack the E7000
Check the components against the component list
Component list
Install the optional LAN board, emulation memory board, or bus monitor interface board
Each board's user's manual
Connect the emulator pod to the emulator station
Section 3.2.1
Connect the external probe
Section 3.2.2
Connect the user system interface cable
Each user system interface cable manual
Install the crystal oscillator
Section 3.2.3
Connect the system ground
Section 3.2.4
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 computer 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
Section 3.4

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 via two station pod interface cables, or to disconnect it when moving the E7000:

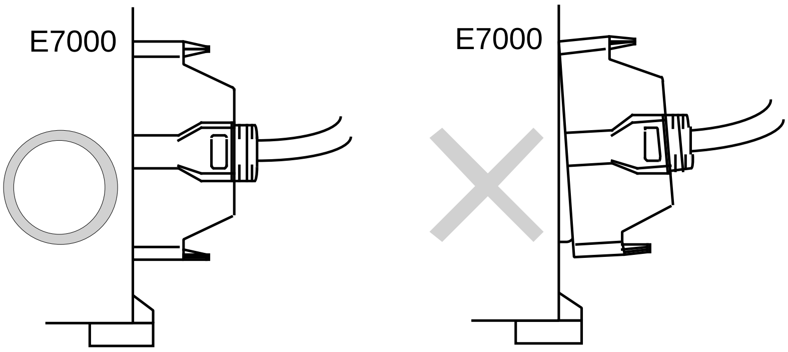
> ⚠ WARNING
>
> **Always switch OFF the emulator product and the user system before connecting or disconnecting the EMULATOR STATION CABLES or the EMULATOR POD. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.**

(1) 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.

⚠ WARNING

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



(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.

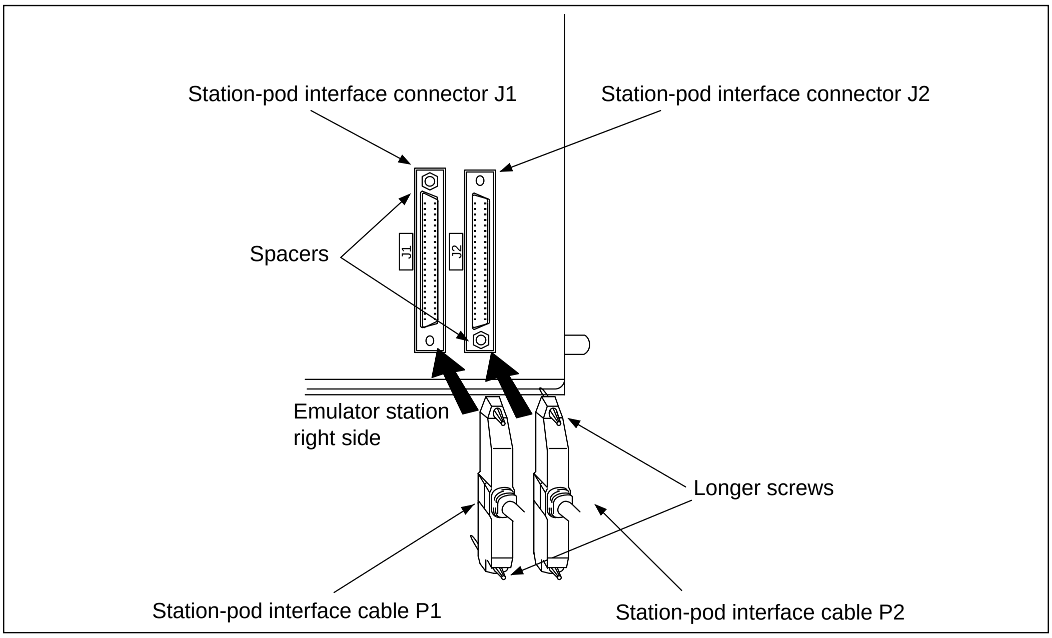

Station-pod interface connector J1
Station-pod interface connector J2
Spacers
J1
J2
Emulator station
right side
Longer screws
Station-pod interface cable P1
Station-pod interface cable P2

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 on the emulator pod, 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 3-3 shows how to connect the station-pod interface cables to the emulator pod.

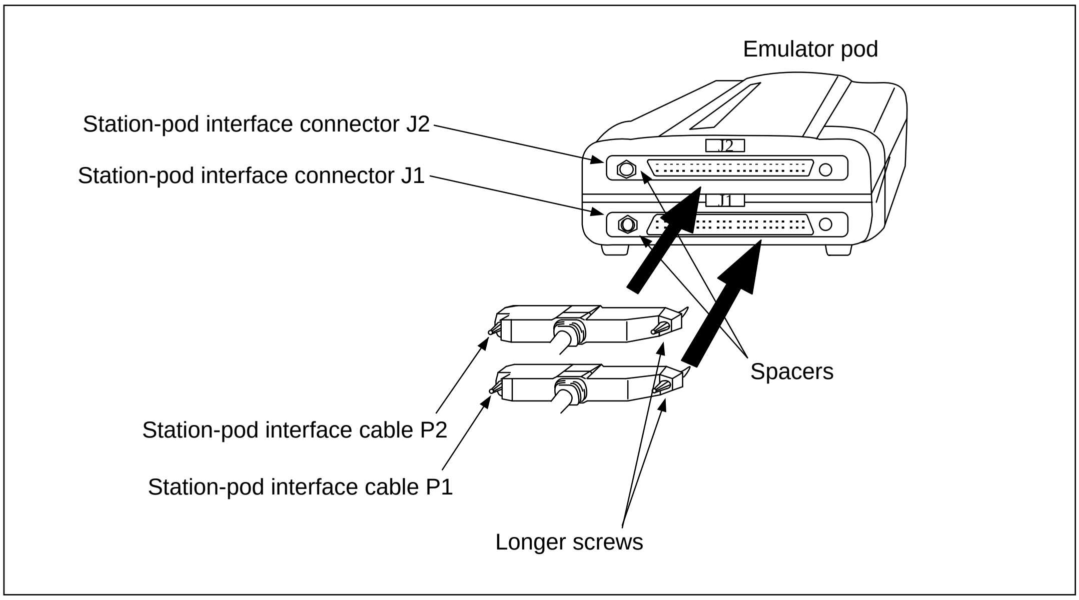

Emulator pod
Station-pod interface connector J2
Station-pod interface connector J1
J2
J1
Spacers
Station-pod interface cable P2
Station-pod interface cable P1
Longer screws

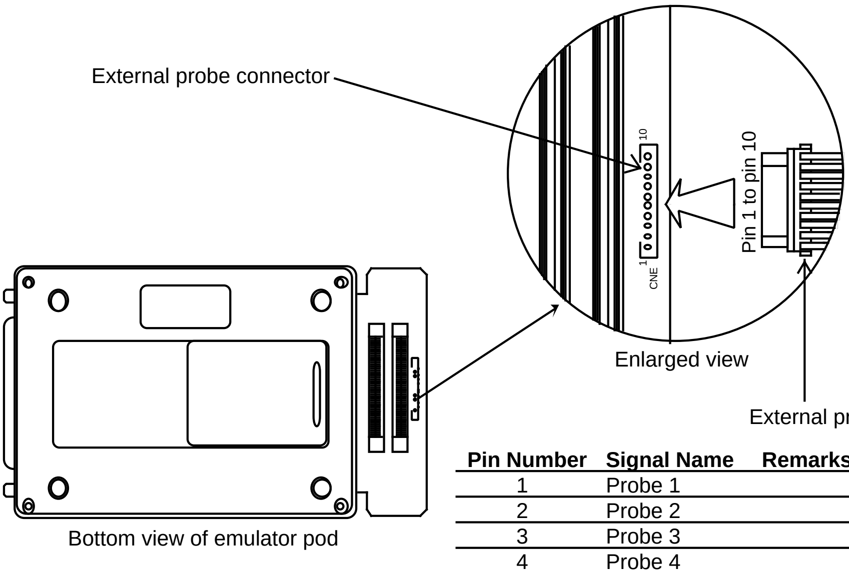
Figure 3-3 Connecting Station-Pod Interface Cables to Emulator Pod

3.2.2 External Probe Connector

CAUTION

Check the external probe direction and connect the external probe to the emulator pod correctly. Incorrect operation will damage the emulator pod and the external probe.

When an external probe is connected to the emulator pod, it enables external signal trace, break using an external signal, and multibreak detection. Figure 3-4 shows the external probe connection. Check the external probe direction and connect the external probe to the emulator pod correctly.



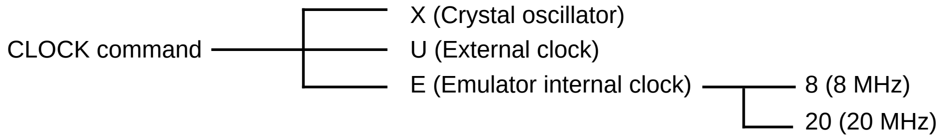
Pin Number	Signal Name	Remarks
1	Probe 1	
2	Probe 2	
3	Probe 3	
4	Probe 4	
5	Probe 5	
6	Probe 6	
7	Probe 7	
8	Probe 8	
9	GND	
10	Tirgger output	Output in trigger mode

Figure 3-4 Connecting External Probe

For connecting a user system interface cable, refer to the user system interface cable manual.

3.2.3 Selecting Clock

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



Crystal Oscillator: A crystal oscillator is not supplied with the emulator. Use one having the frequency used on the user system (ϕ clock frequency). The frequency of the crystal oscillator used as the MCU input clock is 8 to 20 MHz. To use a frequency outside this range, supply an external clock from the user system.

WARNING

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

Use the following procedure to install the crystal oscillator:

1. Check that the emulator power is off.

2. Install the crystal oscillator into the terminal at the bottom of the emulator pod (figure 3-5).

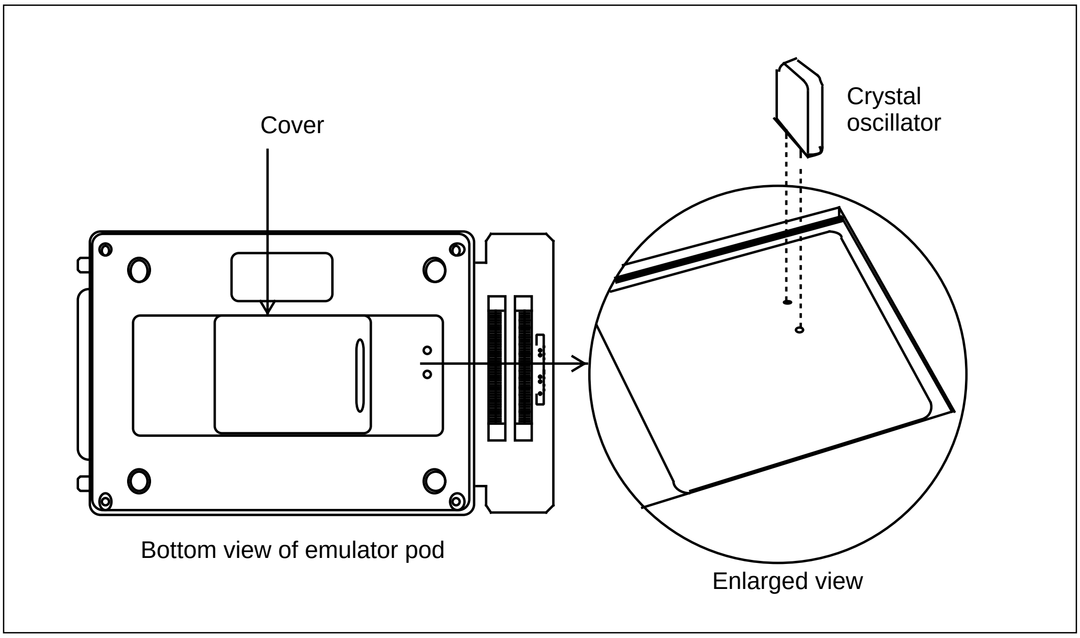


Figure 3-5 Installing the Crystal Oscillator

3. After turning on the emulator, specify X with the CLOCK command.

Using the crystal oscillator enables the user program to be executed at the user system operating frequency, even when the user system is not connected.

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

WARNING

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

1. Check that the emulator power is off.

2. Connect the emulator pod to the user system and supply a clock through the CKIO or EXTAL pin at the end of the user system interface cable from the user system.

3. After turning on the emulator, specify U with the CLOCK command.

Emulator Internal Clock: Specify 8 (8 MHz) or 20 (20 MHz) with the CLOCK command.

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

1. External clock (U) when supplied from the user system
2. Crystal oscillator (X) when installed in the emulator pod
3. 8-MHz emulator internal clock

3.2.4 Connecting SystemGround

⚠ WARNING

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

The E7000 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 3-6). 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 power supply (figure 3-7) so that the ground voltages become the same.

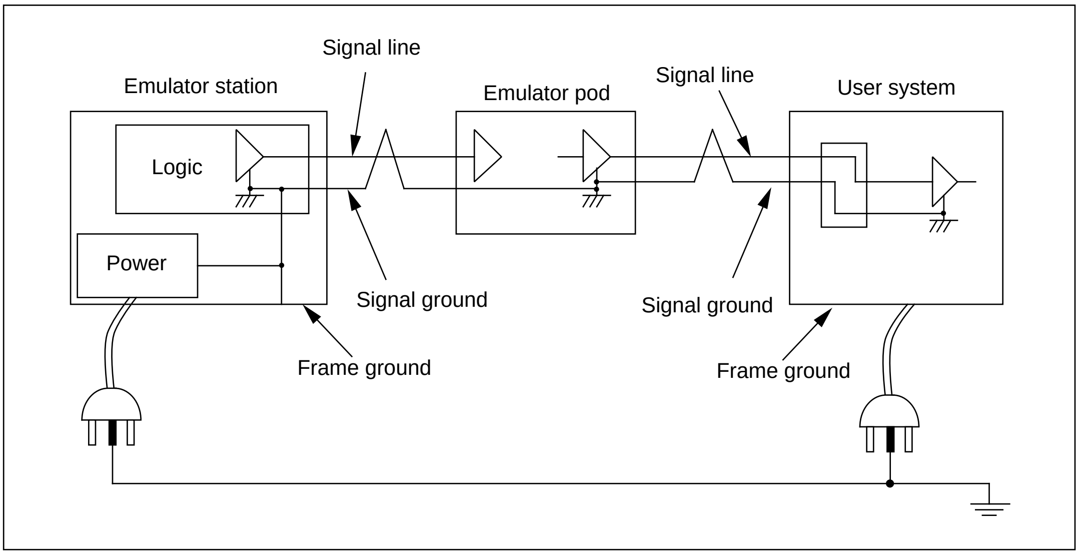


Figure 3-6 Connecting System Ground

⚠ WARNING

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

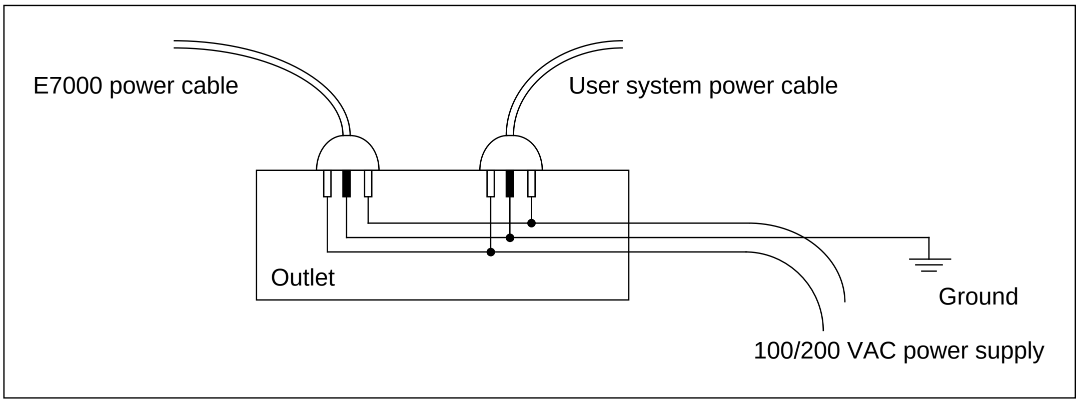


Figure 3-7 Connecting Frame Ground

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 ground.

3.3 System Connection

⚠ WARNING

Ground any apparatus frame at the same outlet as the E7000 emulator station power supply. Failure to do so will result in ELECTROCUTION or will result in PERSONAL INJURY

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

⚠ WARNING

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

The 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 3-8. When a switch is pushed to the right, it turns on, and when it is pushed to the left, it turns off.

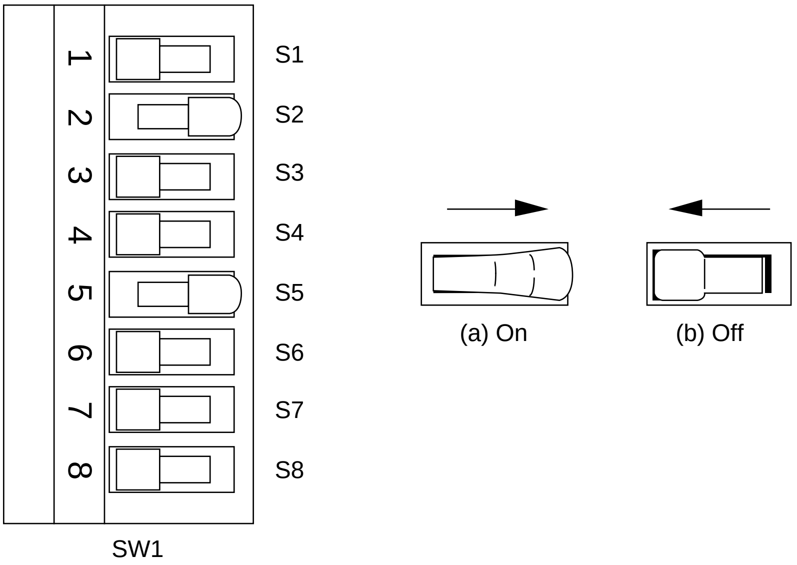


Figure 3-8 Console Interface Switches

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

Note: * indicates setting at shipment.

Number of stop bits

Stop Bits	S4
1 bit	Off*
2 bits	On

Note: * indicates setting at shipment.

Number of data bits

Data Bits	S5
7 bits	Off
8 bits	On*

Note: * indicates setting at shipment.

Parity

Parity	S6
None	Off*
Even/odd	On

Note: * indicates setting at shipment.

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

Parity	S7
Even	Off*
Odd	On

Note: * indicates setting at shipment.

Console/LAN interface selection

Interface	S8
Console	Off*
LAN	On

Note: * indicates setting at shipment.

3.3.3 Connecting to a Host Computer

⚠ WARNING

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

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

The host computer connector (marked HOST) located on the emulator station rear panel conforms to the RS-232C specifications (table 3-3). Connecting a host computer to this connector enables data transfer between the E7000 and the host computer.

Table 3-3 Host Computer Interface Specifications

Item	Specifications
Signal level	RS-232C High: +5 to +15 V Low: –5 to –15 V
Transfer rate	2400/4800/9600/19200/38400 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 Computer Interface Settings at E7000 Start-up: When the E7000 is turned on, or when the E7000 system program is initiated, the host computer 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 Computer 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 computer connector pin assignments and signal names, refer to section B.2, Host Computer Connector. For connection of optional host computer interface cable, refer to section B.6, Host Computer Interface Cable Connection.

3.3.4 Connecting to a Printer

⚠ WARNING

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

The 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

⚠ WARNING

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

The 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 Datagram 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

⚠ WARNING

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

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 3-9 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.

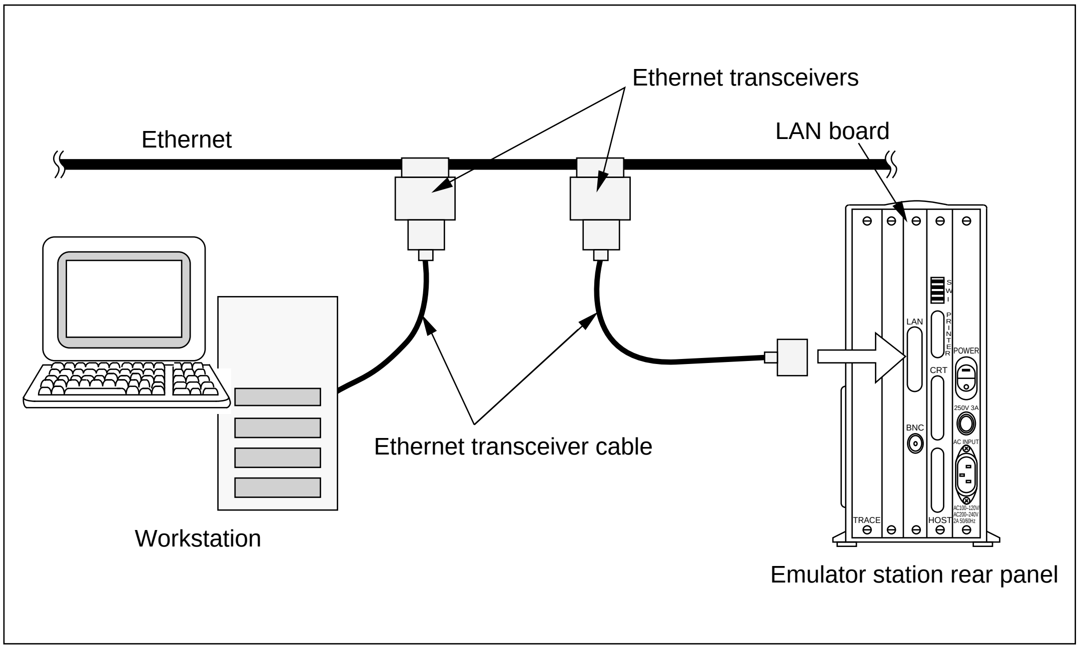


Figure 3-9 Ethernet Interface

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 3-10 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.

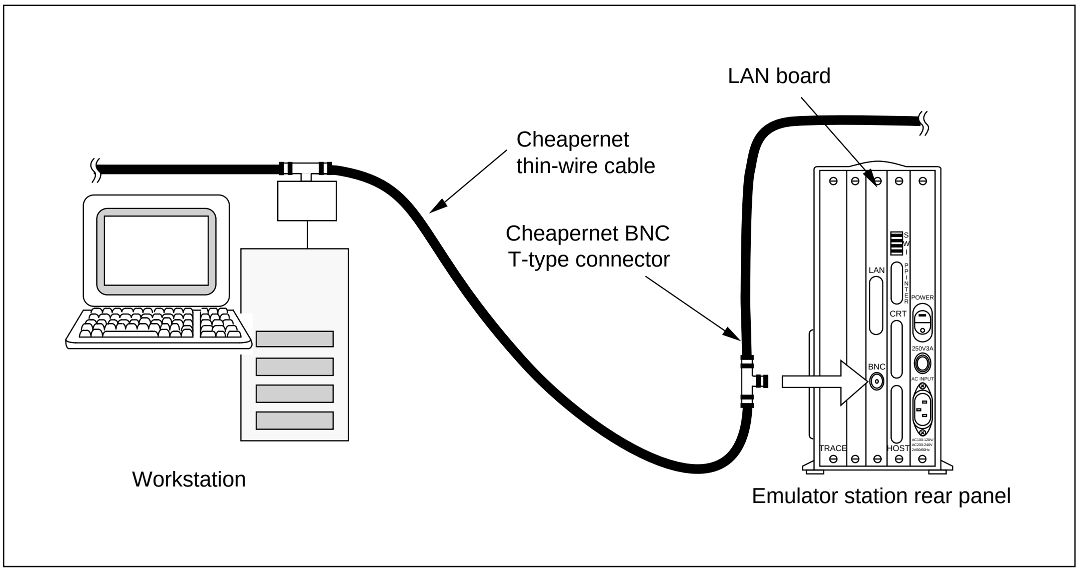


Figure 3-10   Cheapernet Interface

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 3-11.

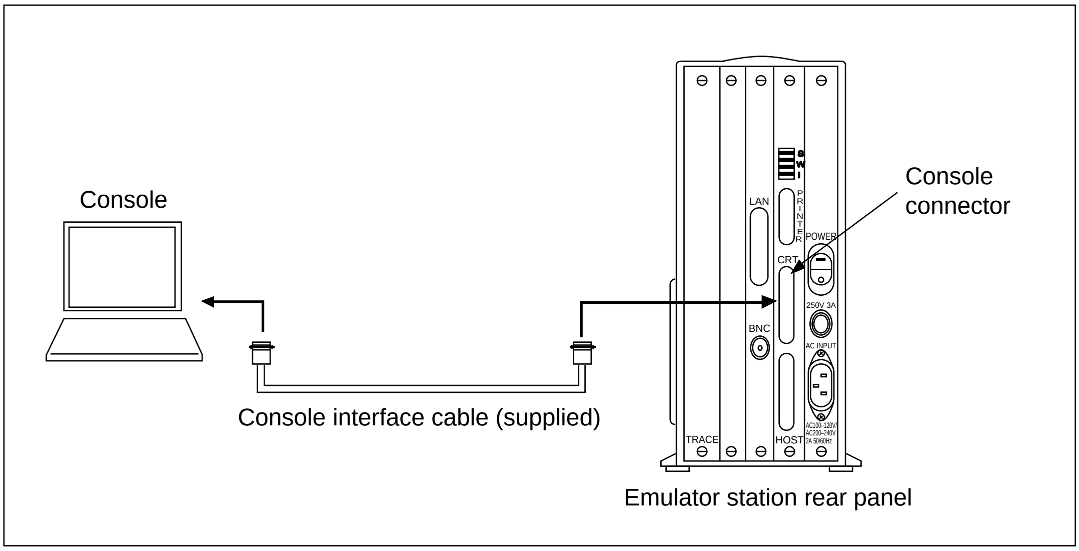


Figure 3-11 Connection in Stand-Alone Mode

Transparent Mode: A console and host computer are connected to the E7000 as shown in figure 3-12.

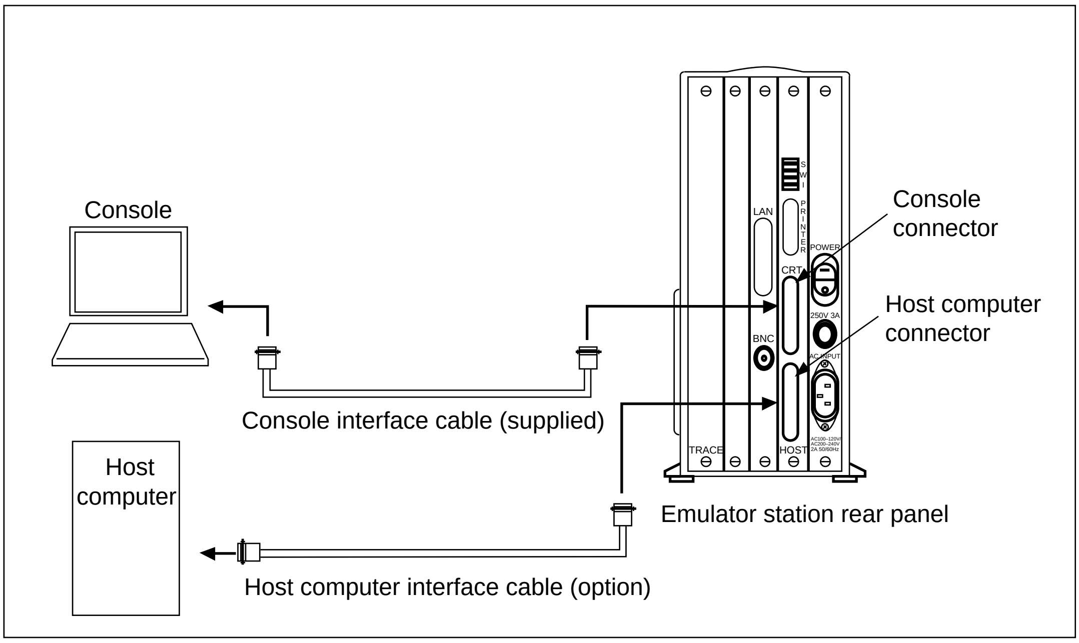

Console
Console connector
Host computer connector
LAN
CRT
BNC
POWER
250V 3A
AC INPUT
TRACE
HOST
Console interface cable (supplied)
Host computer
Emulator station rear panel
Host computer interface cable (option)

Figure 3-12 Connection in Transparent Mode

Local Mode: A console and host computer are connected to the E7000 as shown in figure 3-13.

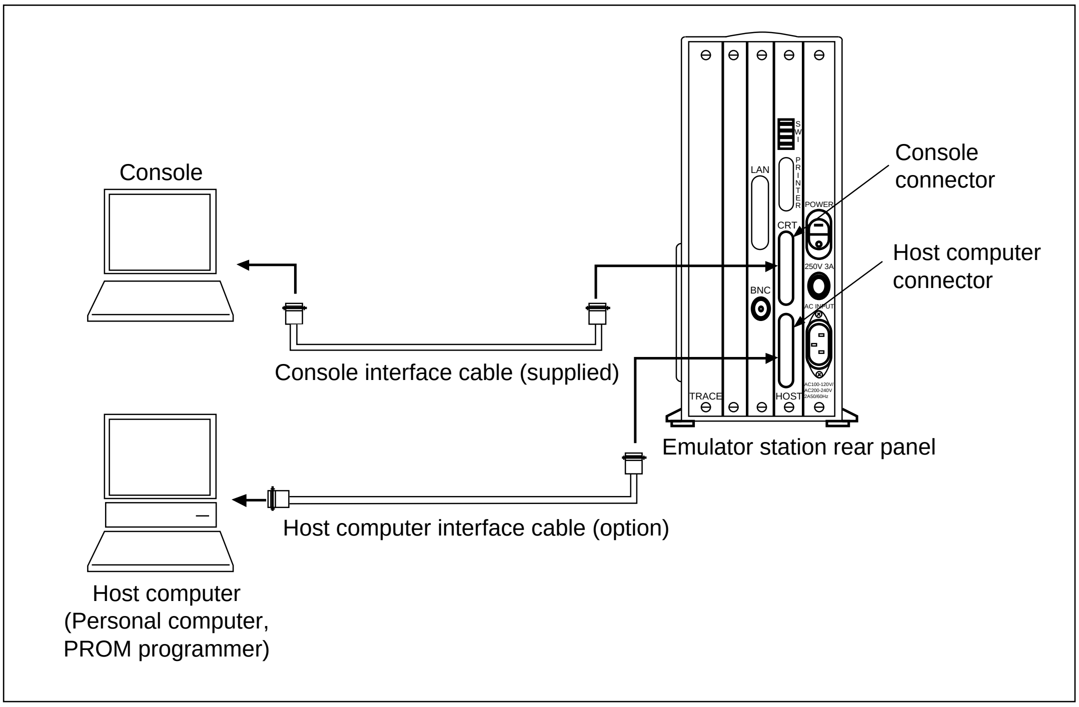


Figure 3-13 Connection in Local Mode

Remote Mode: A host computer is connected to the E7000 as shown in figure 3-14.
The control method in remote mode is X-ON/X-OFF.

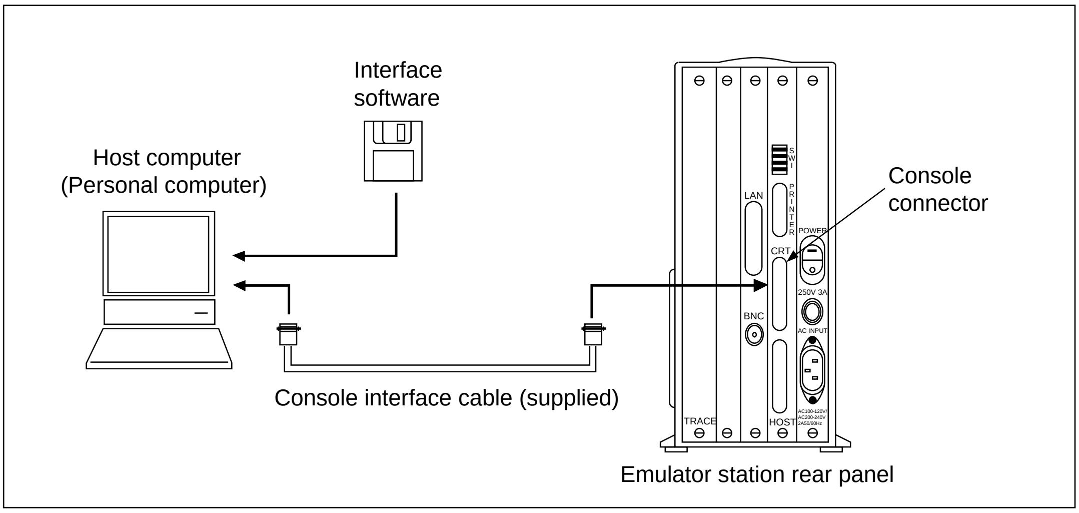


Figure 3-14 Connection in Remote Mode

Printer: A printer is connected to the E7000 as shown in figure 3-15.

WARNING

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

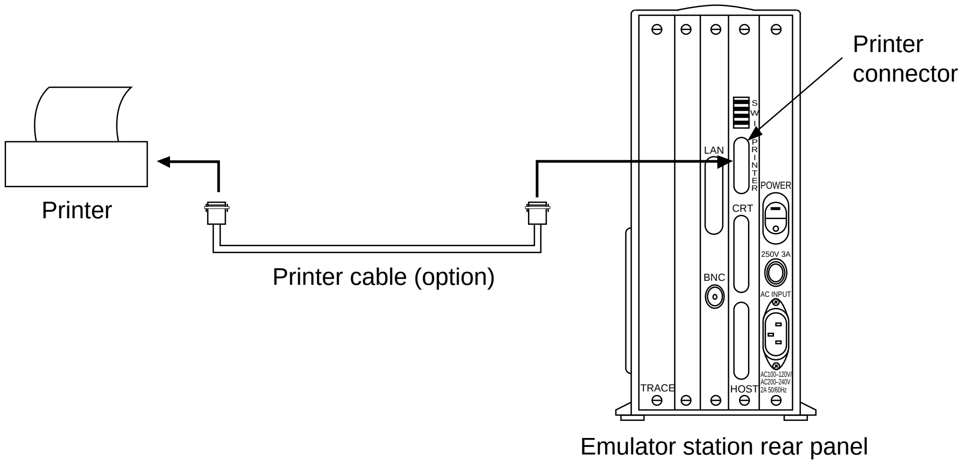


Figure 3-15 Printer Connection

3.4 Power-On and Power-Off Procedures for the E7000

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.When using an RS-232C interface, refer to section 3.4.2, Power-On Procedure for RS-232C Interface.

3.4.1 Power-On Procedure for LAN Interface

Figure 3-16 shows the power-on procedure when the LAN interface is used.

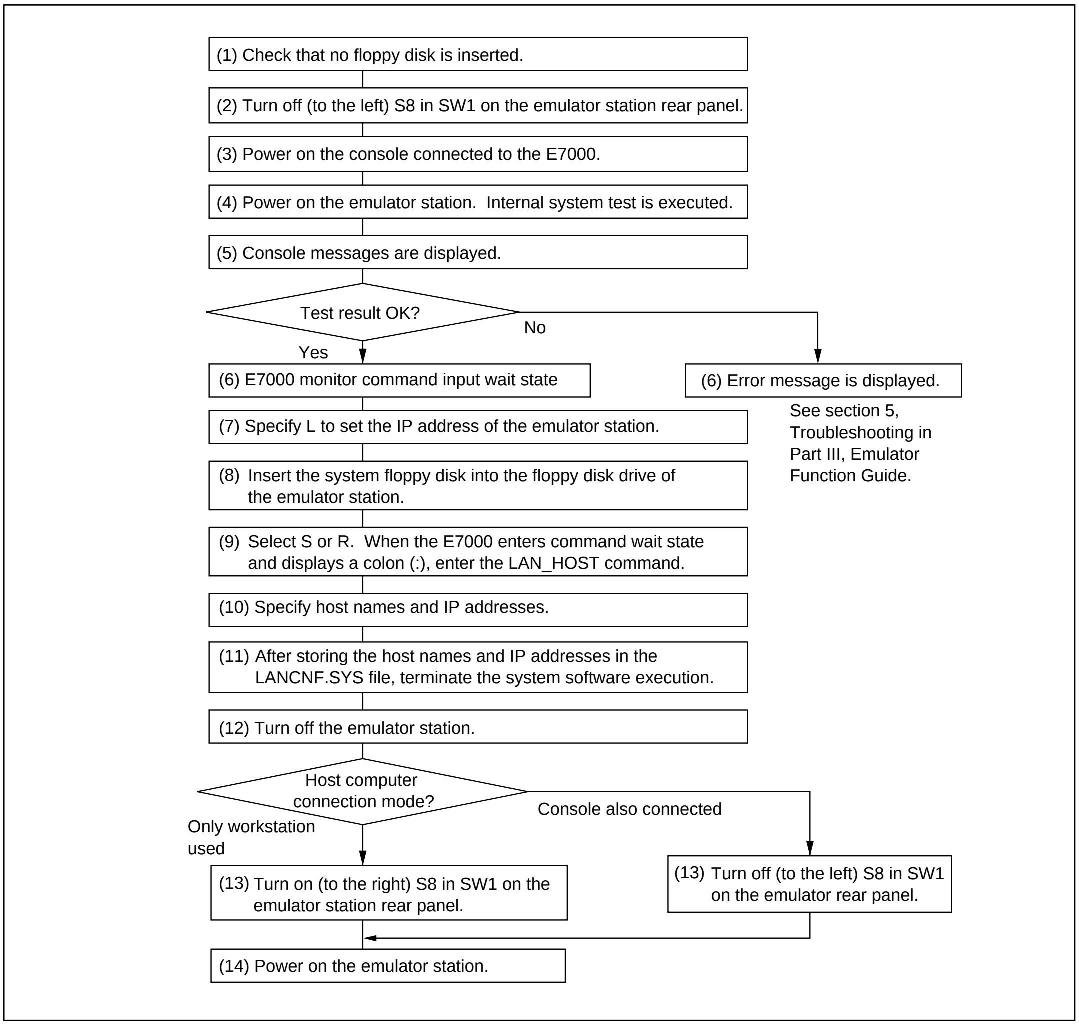

(1) Check that no floppy disk is inserted.
(2) Turn off (to the left) S8 in SW1 on the emulator station rear panel.
(3) Power on the console connected to the E7000.
(4) Power on the emulator station. Internal system test is executed.
(5) Console messages are displayed.
Test result OK?
No
Yes
(6) E7000 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 in the LANCNF.SYS file, terminate the system software execution.
(12) Turn off the emulator station.
Host computer 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 3-16 Power-On Procedure for LAN Interface

The following describes the power-on procedure shown in figure 3-16.

Steps (1) to (7):

The optional LAN board supports the TCP/IP protocol. When the host computer 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. If a floppy disk is inserted, remove it. Check that the cables are firmly connected. 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 computer name and IP address of the E7000 must be specified in the network database for the host computer. When the host computer uses UNIX*, the host name and IP address should generally be specified in the /etc/hosts file. For details, refer to the host computer user’s manual.

Note: UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

Steps (8) to (12):

To transfer data between the host computer and the E7000, initiate the FTP server to connect the host computer to the E7000. Before the FTP server is initiated, the host name and the IP address of the host computer 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 computer> (RET)
```

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

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

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

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

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

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

The specified host names and IP addresses are registered in the LANCNF.SYS file in the E7000 system disk.

Once they have been stored, remove the E7000 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 computer 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 a (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 Commands, 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 3-17 shows the power-on procedure when the RS-232C interface is used.

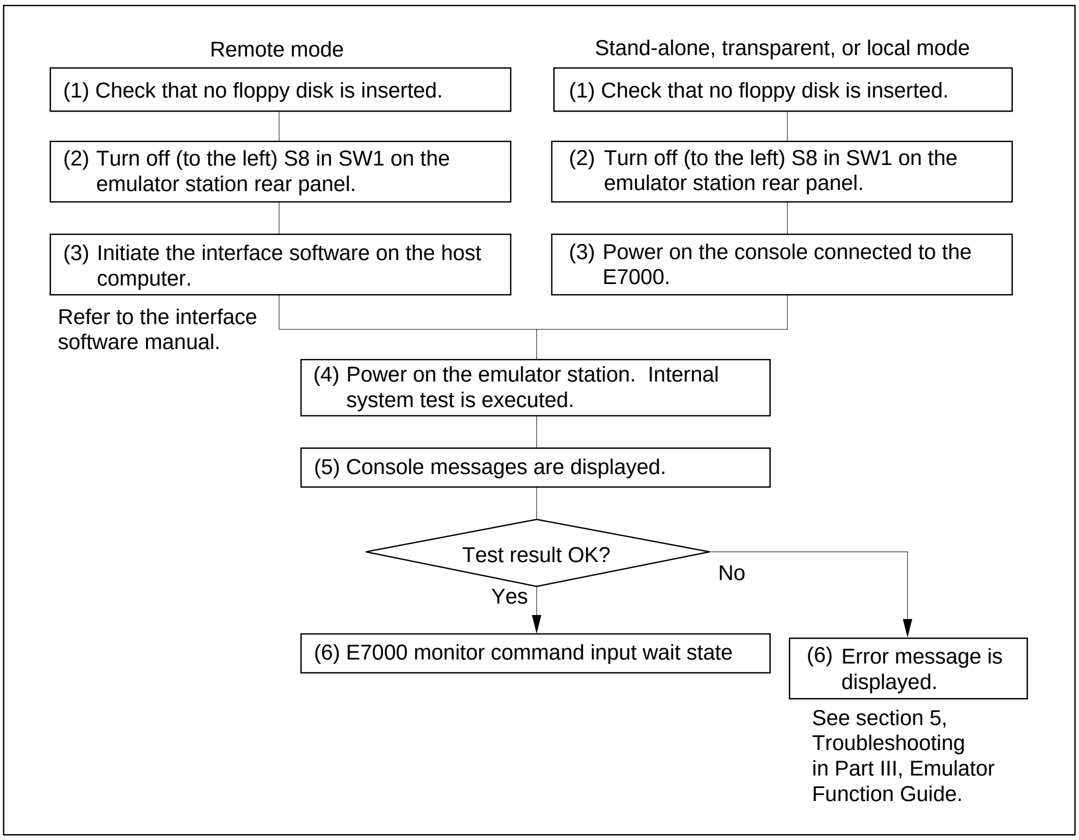

Remote mode
Stand-alone, transparent, or local mode
(1) Check that no floppy disk is inserted.
(1) Check that no floppy disk is inserted.
(2) Turn off (to the left) S8 in SW1 on the emulator station rear panel.
(2) Turn off (to the left) S8 in SW1 on the emulator station rear panel.
(3) Initiate the interface software on the host computer.
(3) Power on the console connected to the E7000.
Refer to the interface software manual.
(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.

Figure 3-17 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.

3.4.3 Power-Off Procedure

Be sure to check that the FTP connection is terminated before turning off the emulator power supply. Otherwise, FTP connection will not work properly.

For details on FTP connection, refer to section 10, Data Transfer from Host Computer Connected by LAN Interface, in Part III, Emulator Function Guide.

CAUTION

Be sure to terminate FTP connection using the BYE command before turning off the power supply. Do not power off the emulator while the floppy disk is being accessed.

To use the current settings when turning on the E7000 the next time, save the settings with the QUIT command before turning off the E7000. For details on the QUIT command, refer to section 7.2.34, QUIT, in Part III, Emulator Function Guide.

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                               (d)
  F: FORMAT FD
  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): (RETURN) key input

S		
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 **
```

B		
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 to another floppy disk 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 disk 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 source floppy disk with the target 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)
```

L		
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 computer 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) ?
```

T

3.5.7	**T**	**Initiates the diagnostic program**

Command Format

- Initiation : T (RET)

Description

- Initiation

  Loads the diagnostic program from the E7000 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.

> CAUTION
>
> **1. Only use 2HD (double sided, high density, double track) floppy disks.**
>
> **2. Do not remove the disk while the disk drive is operating or the floppy disk contents will be lost.**

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 lists the E7000 system program initiation commands.

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)

H8S/xxxx E7000 (HSxxxx EPDxxSF) Vn.m
Copyright (C) Hitachi, Ltd. 199x                        (c)
Licensed Material of Hitachi, Ltd.
```

```
LAN IP ADDRESS FILE LOADING                                                 (d)
CONFIGURATION FILE LOADING                                                  (e)
HARD WARE REGISTER READ/WRITE CHECK                                         (f)
POD SYSTEM LOADING                                                          (g)
EMULATOR POD TEST                                                           (h)
**  RESET IN BY E7000  !                                                    (i)
CLOCK = xx MHz                                                              (j)
CPU MODE=H8S/xxxx  OPERATION MODE=x(MDx-x=x)  MODE SET=xxxx
PIN MODE=xxx/xxx  INTERNAL ROM SIZE=xxxxB  INTERNAL RAM SIZE=xxB
ADC MODE=xxxxxx  DMAC MODE=xxxxxx  REFRESH MODE=xxxxxx                      (k)
SCI CHANNEL=xCH  16BIT TIMER MODE=TPUx-x
REMAINS EMULATION MEMORY      S=xxxxx /E=xxxxxx                             (l)

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

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) IP address file for the LAN is being loaded.

(e) Configuration file is being loaded.

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

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

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

(i) A RES signal has been input to the MCU.

(j) Specified clock. If the user system is ready, the user system’s clock is used. If the user system’s clock is not ready, the crystal oscillator in the emulator pod is used. If the crystal oscillator is not ready, the 8-MHz E7000 internal clock is used.

(k) MCU type, MCU operating mode, and number of pins. They are previously set with the MODE command (saved in configuration file). For details, refer to section 7.2.29, MODE, in Part III, Emulator Function Guide.

(l) Remaining emulation memory size

(m) 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.

(n) 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 7.2.34, QUIT.) The recovery information is listed below.

- **Software breakpoints**
- **Hardware break conditions, and trace stop and acquisition conditions**
- **Memory map information**
- **Performance analysis 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 the E7000 internal system test at power-on has been completed, the E7000 system program is automatically loaded.

```
Console display                                              Procedure

E7000 MONITOR Vn.m                                           After E7000 power-on,
Copyright (C) 1991 Hitachi, Ltd.                             insert the system disk when
Licensed Material of Hitachi, Ltd.                           this message is displayed.

TESTING
 RAM      0123

**  E7000 SYSTEM LOADING  **                                 E7000 system program
                                                             is being loaded.

H8S/xxxx E7000  (HSxxxx EPDxxSF)  Vn.m                       System program is initiated.
Copyright (C) Hitachi, Ltd. 199x                             (Refer to section 3.5,
Licensed Material of Hitachi, Ltd.                           E7000 Monitor Commands.)

LAN IP ADDRESS FILE LOADING
CONFIGURATION FILE LOADING
HARD WARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TEST
**  RESET IN BY E7000 !
CLOCK = xx MHz
CPU MODE=H8S/xxxx  OPERATION MODE=x(MDx-x=x)  MODE SET=xxxx
PIN MODE=xxx/xxx  INTERNAL ROM SIZE=xxxxB  INTERNAL RAM SIZE=xxB
ADC MODE=xxxxxx  DMAC MODE=xxxxxx  REFRESH MODE=xxxxxx
SCI CHANNEL=xCH  16BIT TIMER MODE=TPUx-x
REMAINS EMULATION MEMORY      S=xxxxx /E=xxxxxx

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

Section 4   Operating Examples

Section 4.1, Basic Examples, and section 4.2, Application Examples, include explanations on how to operate the E7000 based on the following user programs.

```
ADDR      CODE       LABEL    MNEMONIC    OPERAND
000100    7A0700FF             MOV.L      #00FFF000:32,ER7
          F000
000106    F800                 MOV.B      #00:8,R0L
000108    F900                 MOV.B      #00:8,R1L
00010A    8802                 ADD.B      #02:8,R0L
00010C    8901                 ADD.B      #01:8,R1L
00010E    A90A                 CMP.B      #0A:8,R1L
000110    46F8                 BNE        00010A:8
000112    6A881000             MOV.B      R0L,@1000:16
000116    40FE                 BRA        000116:8
```

These examples assume that the emulator station is connected to the LAN host computer with the TELNET and that the user program is downloaded from the host computer to the E7000. Therefore, store the program in the host computer before initiating the E7000. In these examples, the host name is HITACHI, and the IP address is 128.1.1.10.

Initiate the E7000 by the following procedure:

Note: Only parts of the console message are shown depending on the amount of vacant space in this document.

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 **

H8S/xxxx E7000 (HSxxxxEPDxxSF) Vn.m
Copyright (C) Hitachi, Ltd. 199x
Licensed Material of Hitachi, Ltd.

LAN IP ADDRESS FILE LOADING
CONFIGURATION FILE LOADING
HARD WARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TESTING
** RESET IN BY E7000 !
CLOCK = xx MHz
CPU MODE=H8S/xxxx  OPERATION MODE=x(MD x-x
PIN MODE=xxx/xxx  INTERNAL ROM SIZE=xxxxB  IN
ADC MODE=xxxxxx  DMAC MODE=xxxxxxx  REFRES
SCI CHANNEL=xCH  16BIT TIMER MODE=TPUx-x
REMAINS EMULATION MEMORY S=xxxxx/E=xxxxxx

WARM OR COLD START
   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 Computer

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

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

Operations | **Console Message**

1. Specify the IP address of the host computer to which the E7000 is to be connected by the FTP command. Enter LAN_HOST;S (RET), and the console will display the host computer 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         128.1.1.1   02       HOST_B         128.1.1.2
03       HOST_C         128.1.1.3   04       HOST_D         128.1.1.4
05       HOST_E         128.1.1.5   06       HOST_F         128.1.1.6
07       HOST_G         128.1.1.7   08       HOST_H         128.1.1.8
09       HOST_I         128.1.1.9
PLEASE SELECT NO? _
```

2. Enter 1 (RET) as the selection number, HITACHI (RET) as the host computer name, and 128.1.1.10 (RET) as the IP address. After that, the console prompts the user to select another number. Enter . (RET) to exit interactive mode.
   After the host computer 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   128.1.1.1    ?  128.1.1.10  (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.

```
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)
```

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

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

5. Enter (RET).

4.1.2 Specifying the MCU Operating Mode

Specify the E7000 operating mode and the MCU operating mode by the following procedure:

> ⚠ WARNING
>
> **Correctly specify the operating mode according to the TARGET DEVICE settings (table 4-1) . The emulator will not operate correctly with an incorrect setting. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.**

In the following example, MCU operating mode 4 of the H8S/2653 is selected. When using other target devices, specify the operating mode according to the target device settings shown in table 4-1. In the following console message example, x indicates the current (before-change) value.

Table 4-1 Target Device Settings

	Target Device		
Item	**H8S/2653**	**H8S/2655**	**H8S/2245**
CPU MODE	2	2	1
OPERATION MODE	1 to 7	1 to 7	1 to 7
MODE SET	1 or 2	1 or 2	1 or 2
PIN MODE	4	4	6
INTERNAL ROM SIZE	3 (64 kbytes)	5 (128 kbytes)	5 (128 kbytes)
INTERNAL RAM SIZE	4 (4 kbytes)	4 (4 kbytes)	4 (4 kbytes)
ADC MODE	2	2	1
DMAC MODE	2	2	1
REFRESH MODE	2	2	1
SCI CHANNEL	2	2	2
16BIT TIMER MODE	2	2	1

Operations **Console Message**

1. Remove the write protect from the system floppy disk before storing the mode settings in the configuration file.

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

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

3. The console displays the message shown on the right. To select the H8S/2653 as the target MCU, for example, enter 2 (RET).

```
CPU MODE(1:H8S/2000, 2:H8S/2600) x ? _

CPU MODE(1:H8S/2000, 2:H8S/2600) x ? 2 (RET)
```

4. To select mode 4 as the MCU operating mode, for example, enter 4 (RET).

```
OPERATION MODE  (MD2-0) x ? 4 (RET)
```

5. The method for setting the operating mode is specified as follows:
   1: Selects the operating mode set in the above step
   2: Selects the operating mode set by the operating mode selection pin (MD2 to MD0) on the user system
   To use the emulator without connecting the user system, for example, enter 1 (RET).

```
MODE SET(1:E7000 MODE, 2:USER MODE) x ? 1 (RET)
```

6. To select the TFP-120 package as the number of pins, for example, enter 4 (RET).

```
PIN MODE (1:80/84-A, 2:100-A, 3:112, 4:120/128, 5:80/84-B, 6:100-B) x ? 4 (RET)
```

7. To select 64 kbytes as the internal ROM size because the H8S/2653 has 64-kbyte ROM, enter 3 (RET).

```
INTERNAL ROM SIZE (1:0KB, 2:32KB, 3:64KB, 4:96KB, 5:128KB, 6:256KB) x ? 3 (RET)
```

8. To select 4 kbytes as the internal RAM size because the H8S/2653 has 4-kbyte RAM, enter 4 (RET).

```
INTERNAL RAM SIZE (1:1KB, 2:2KB, 3:3KB, 4:4KB, 5:6KB, 6:8KB) x ? 4 (RET)
```

9. Select the internal AD converter.

```
ADC MODE (1:NORMAL, 2:HI SPEED) x ? 2 (RET)
```

10. Select the internal DMAC.

```
DMAC MODE (1:DISABLE, 2:ENABLE) x ? 2 (RET)
```

11. Select whether to enable the internal refresh controller.

```
REFRESH MODE (1:DISABLE, 2:ENABLE) x ? 2 (RET)
```

12. Select the number of SCI channels.

```
SCI CHANEL (1:2CH, 2:3CH) x ? 2 (RET)
```

13. Select a combination of 16-bit timers.

```
16BIT TIMER MODE (1:TPU0-2, 2:TPU0-5) x ? 1 (RET)
```

14. 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 2 above before entering Y (RET) and repeat the procedure.

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

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

15. 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) ?  _
```

16. 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) ? _
```

17. 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 and execute it, allocate the standard emulation memory in the emulator pod by the following procedure:

Operations

Enter MAP 0 1FFFF;S (RET) to allocate the standard emulation memory to addresses H'0 to H'1FFFF. The console displays the message shown on the right, which indicates that the memory allocation has been completed.
Enter MAP (RET) and the console displays the attributes of all the memory areas.

Console Message

```
:MAP 0 1FFFF;S (RET)


 REMAINS EMULATION MEMORY     S=60000/E=000000


:MAP (RET)
 00000000-0001FFFF;S   00020000-00FFFFFF;U
 INTERNAL RAM  =  00FFEC00-00FFFBFF
 INTERNAL I/O      =  00FFFE40-00FFFFFF
 REMAINS EMULATION MEMORY     S=60000/E=000000
:
```

4.1.4 Loading the User Program

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

Operations	Console Message
1. Enter FTP <host name> (RET) to connect the E7000 and the host computer 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 computer 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 actual address to which the program is being loaded instead of xxxxxx.	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

Caution

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 to execute the BYE command before powering down the E7000. If the E7000 is powered down before executing the BYE command, the FTP connection is disconnected.

8. The DISASSEMBLE command displays the loaded program. Enter DISASSEMBLE 100 117 (RET).

```
:DISASSEMBLE 100 117 (RET)
 ADDR          CODE        LABEL      MNEMONIC       OPERAND
 00000100      7A0700FF                MOV.L         #00FFF000:32,ER7
               F000
 00000106      F800                    MOV.B         #00:8,R0L
 00000108      F900                    MOV.B         #00:8,R1L
 0000010A      8802                    ADD.B         #02:8,R0L
 0000010C      8901                    ADD.B         #01:8,R1L
 0000010E      A90A                    CMP.B         #0A:8,R1L
 00000110      46F8                    BNE           00010A:8
 00000112      6A881000                MOV.B         R0L,@1000:16
 00000116      40FE                    BRA           000116:8
```

4.1.5 Executing Program

Execute the loaded program by the following procedure:

Operations **Console Message**

1. Set the initial values of the registers. Enter .SP (RET) then FFF000 (RET) as the SP value to set the stack pointer (SP register) to H'FFF000. The console then asks for the program counter value. Enter 100 (RET) as the program counter value. The console then asks for the condition code register value. In this example, other registers need not be set or changed, therefore, enter . (RET) to exit this interactive mode.

```
:.SP (RET)
 ER7(SP) = 00FFF7FE ? _

 ER7(SP) = 00FFF7FE ? FFF000 (RET)
 PC         = 00FFFFFF ? _

 PC         = 00FFFFFF ? 100 (RET)
 CCR      = 80:I****** ? _

 CCR      = 80:I****** ?  . (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 .(RET) to the condition code register prompt can complete the register modification procedure. A register value can also be directly input without using the interactive mode. For example, to set the stack pointer value directly, enter .SP FFF000 (RET).

2. Enter GO (RET) to execute the loaded program from the address pointed to 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 registers such as the program counter, the condition code register, and the general registers ER0 to ER7 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 keys were entered.

```
:(BREAK)

 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D'0000H:00M:01S:049705US
 +++: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 10C (RET) to terminate program execution immediately before the instruction at address H'10C in the program has been executed.

```
:BREAK 10C (RET)
```

2. Restart program execution from address H'100. 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'100, then enter GO, as described in section 4.1.5, Executing Program.

```
:GO 100 (RET)
 **  PC = xxxxxxxx
```

3. The program execution terminates immediately before the instruction at address H'10C has been executed. The console displays the data shown on the right. BREAK POINT 0000010C shows that the program execution was terminated because of a software breakpoint at address H'10C.

```
 PC =0000010C   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000002  00000000  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:00S:000007US
 +++:BREAK POINT  0000010C
:_
```

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. 0000010C ADD.B #01:8,R1L shows the executed address and mnemonic code, and +++:STEP NORMAL END shows that the single-step execution has terminated.

```
:STEP (RET)



 PC =0000010E   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000002  00000001  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 0000010C                  ADD.B         #01:8,R1L
 +++:STEP NORMAL END
:_
```

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

```
:(RET)
 PC =00000110   CCR =A9:I*H*N**C   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000002  00000001  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 0000010E                  CMP.B         #0A:8,R1L
 +++: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'1000, enter BREAK_CONDITION1 A = 1000 W (RET).

```
:BREAK_CONDITION1 A = 1000 W (RET)
```

4. Enter GO 100 (RET) to start executing the program from address H'100. 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 100 (RET)
 **  RUNNING
 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D'0000H:00M:00S:000018US
 +++: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 instruction mnemonic information of the executed instructions.

```
:TRACE  (RET)
```

```
    IP        ADDR       LABEL    MNEMONIC    OPERAND
*-D'00044   00000100              MOV.L       #00FFF000:32,ER7
*-D'00043   00000106              MOV.B       #00:8,R0L
*-D'00042   00000108              MOV.B       #00:8,R1L
*-D'00041   0000010A              ADD.B       #02:8,R0L
*-D'00040   0000010C              ADD.B       #01:8,R1L
    :          :                    :            :
```

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

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

```
    BP          AB      DB    MA    R/W  ST    IRQ       NMI  RA    PROG       CLK
*           00000100               MOV.L     #00FFF000:32,ER7
 -D'00062   00000100  7A07  EXT    R   PRG   11101111   1    11    11111111   07
  -D'00061  00000102  00FF  EXT    R   PRG   11101111   1    11    11111111   06
 -D'00060   00000104  F000  EXT    R   PRG   11101111   1    11    11111111   06
*           00000106               MOV.B     #00:8,R0L
  -D'00059  00000106  F800  EXT    R   PRG   11101111   1    11    11111111   06
*           00000108               MOV.B     #00:8,R1L
  -D'00058  00000108  F900  EXT    R   PRG   11101111   1    11    11111111   06
*           0000010A               ADD.B     #02:8,R0L
 -D'00057   0000010A  8802  EXT    R   PRG   11101111   1    11    11111111   06
*           0000010C               ADD.B     #01:8,R1L
 -D'00056   0000010C   8901 EXT    R   PRG   11101111   1    11    11111111   06
                 :                           :                     :
```

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 10A 5 (RET) to terminate program execution immediately after address H'10A is passed five times.

```
:BREAK  10A  5 (RET)
```

2. To start execution from address H'100, enter GO 100 (RET).

3. When execution terminates after address H'10A is passed five times, the console displays the data shown on the right.

```
:GO 100 (RET)
 **  RUNNING
 PC =0000010A   CCR =A9:I*H*N**C  EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000008  00000004  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D'0000H:00M:00S:000040US
 +++:BREAK POINT 0000010A
:_
```

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>
 0000010A          0005       0005
    (a)                 (b)           (c)
```

4.2.2 Conditional Trace

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

Operations **Console Message**

1. Enter BREAK - (RET) and BREAK_CONDITION1 - (RET) to cancel the breakpoints set in the example of section 4.2.1, Break with Pass Count Condition, and section 4.1.8, Setting Hardware Break Conditions.

```
:BREAK - (RET)
:BREAK_CONDITION1 - (RET)
```

2. Enter TRACE_CONDITION A =100:106;R (RET) to get trace information only while the program counter is between addresses H'100 and H'106.

```
:TRACE_CONDITION A= 100:106;R (RET)
```

3. Enter GO 100 (RET) to start executing the program, then (BREAK) key or (CTRL) + C keys to terminate the execution.

```
:GO 100 (RET)
 **  PC = xxxxxxxx  (BREAK)
 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:01S:430529US
 +++:BREAK KEY
:_
```

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

```
:TRACE  (RET)

     IP          ADDR        LABEL    MNEMONIC     OPERAND
 *-D'*****     00000100               MOV.L        #00FFF000:32,ER7
 *-D'*****     00000106               MOV.B        #00:8,R0L
     :              :                    :                :
```

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 100 (RET)
 ** PC = xxxxxxxx    (RET)
#_
```

2. Enter DUMP 1000 100F (RET) to display the memory contents from H'1000 to H'100F in parallel mode.

```
#DUMP 1000 100F (RET)
```

```
<ADDR>              <    D   A     T   A    >                    <ASCII CODE>
00001000   14  00  00  00  00  00  00  00      00  00  00  00  00  00  00  00      “. . . . . . . .”
```

3. Enter MEMORY 117 EE (RET) to modify the contents of memory address H'117 into EE in parallel mode.

```
#MEMORY 117 EE (RET)
```

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

```
#END (RET)
 ***81: TRACE  CONDITION  RESET
 ** PC = xxxxxxxx
```

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

```
(BREAK)
 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:00S:0430529US
 +++:BREAK KEY
:_
```

6. Enter DISASSEMBLE 100 117 (RET) to confirm that the program has been changed by memory modification in parallel mode.

```
:DISASSEMBLE 100 117 (RET)
```

```
ADDR         CODE          LABEL          MNEMONIC       OPERAND
000100       7A0700FF                      MOV.L         #00FFF000:32,ER7
             F000
000106       F800                          MOV.B         #00:8,R0L
000108       F900                          MOV.B         #00:8,R1L
00010A       8802                          ADD.B         #02:8,R0L
00010C       8901                          ADD.B         #01:8,R1L
00010E       A90A                          CMP.B         #0A:8,R1L
000110       46F8                          BNE           00010A:8
000112       6A881000                      MOV.B         R0L,@1000:16
000116       40EE                          BRA           000106:8
```

4.2.4 Searching Trace Information

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

Operation

Enter TRACE_SEARCH A=116 (RET), and the console will only display those parts of the trace information in which the address bus value is H'116.

Console Message

```
:TRACE_SEARCH A=116 (RET)

      BP          AB       DB    MA   R/W  ST    IRQ       NMI  RA   PROG       CLK
   -D'04077   00000116   40EE  EXT   R   PRG  11101111   1   11   11111111   06
   -D'03964   00000116   40EE  EXT   R   PRG  11101111   1   11   11111111   06
   -D'03851   00000116   40EE  EXT   R   PRG  11101111   1   11   11111111   06
                  :                              :                          :
```

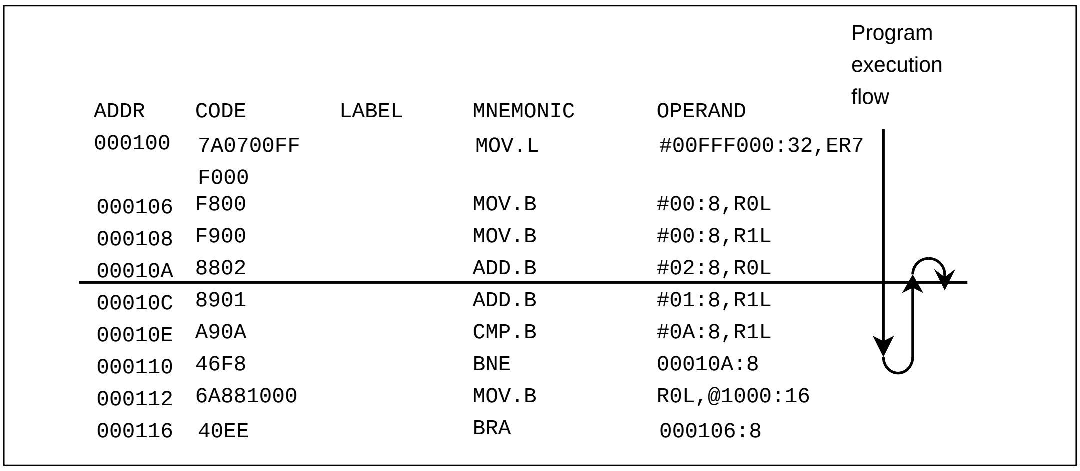
4.2.5 Sequential Software Break

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 110 10A (RET), which terminates program execution when the instructions at addresses H'110 and H'10A are executed consecutively in that order, as shown in figure 4-1.

```
:BREAK_SEQUENCE 100 10A (RET)
```



```
ADDR    CODE       LABEL     MNEMONIC       OPERAND
000100  7A0700FF             MOV.L          #00FFF000:32,ER7
        F000
000106  F800                 MOV.B          #00:8,R0L
000108  F900                 MOV.B          #00:8,R1L
00010A  8802                 ADD.B          #02:8,R0L
00010C  8901                 ADD.B          #01:8,R1L
00010E  A90A                 CMP.B          #0A:8,R1L
000110  46F8                 BNE            00010A:8
000112  6A881000             MOV.B          R0L,@1000:16
000116  40EE                 BRA            000106:8
```

Figure 4-1 Program Execution Flow

2. Enter GO 100 (RET) to execute the program. When the specified condition is satisfied, execution terminates, and the console displays the data shown on the right. +++:BREAK SEQUENCE shows that execution has terminated because the condition specified in the BREAK_SEQUENCE command has been satisfied.

```
:GO 100 (RET)
** PC = xxxxxxxx
PC =0000010A   CCR =A9:I*H*N**C  EXR=07:*****210
MACH=00000000   MACL=00000000
ER0 - ER3   00000002  00000001  00000000  00000000
ER4 - ER7   00000000  00000000  00000000  00FFF000
RUN - TIME = D'0000H:00M:00S:000018US
+++:BREAK SEQUENCE
:_
```

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

```
:TRACE  (RET)
```

```
   IP        ADDR       LABEL   MNEMONIC     OPERAND
*-D'00009   00000100            MOV.L        #00FFF000:32,ER7
*-D'00008   00000106            MOV.B        #00:8,R0L
*-D'00007   00000108            MOV.B        #00:8,R1L
*-D'00006   0000010A            *  BREAK  *
*-D'00005   0000010A            ADD.B        #02:8,R0L
*-D'00004   0000010C            ADD.B        #01:8,R1L
*-D'00003   0000010E            CMP.B        #0A:8,R1L
*-D'00002   00000110            *  BREAK  *
*-D'00001   00000110            BNE          00010A:8
* D'00000   0000010A            *  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.

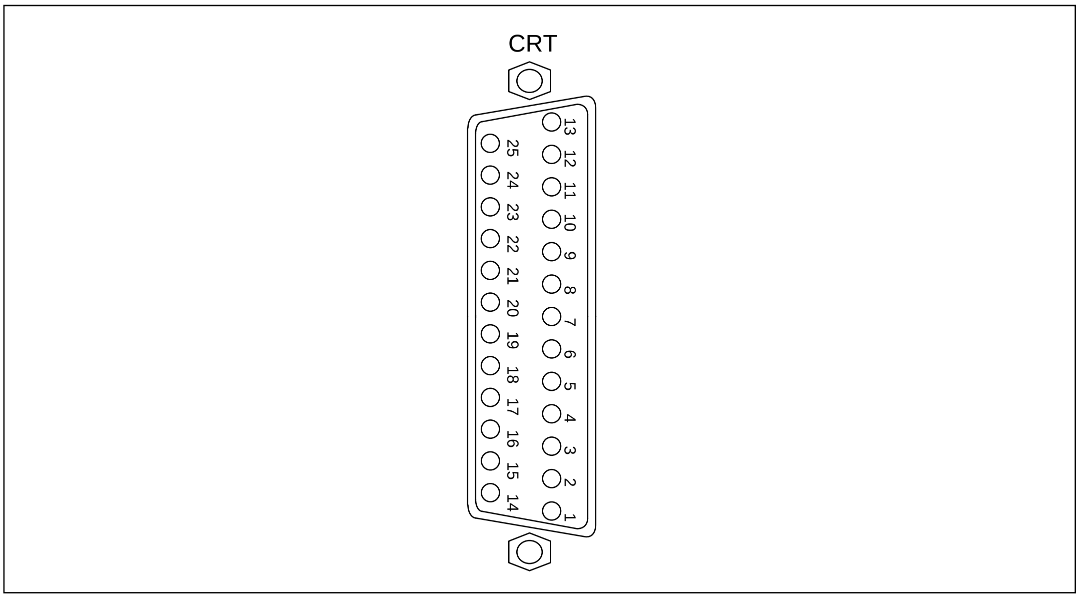


Figure B-1   Console Connector Pin Locations

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 connected
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 to 19	Not connected	—
20	Data Set Ready (DSR)	Not connected
21 to 25	Not connected	—

B.2 Host Computer Connector

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

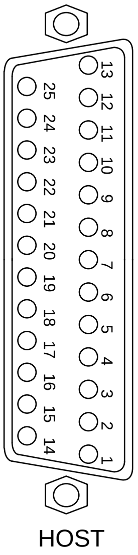


Figure B-2 Host Computer Connector Pin Locations

Table B-2 Signal Names and Usage of Host Computer 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 connected
7	Signal Ground (SG)	Signal ground
8 to 19	Not connected	—
20	Data Terminal Ready (DTR)	High when E7000 power is on
21 to 25	Not connected	—

B.3 Printer Connector

Figure B-3 shows pin locations in the connector at the E7000 end of the provided printer cable. Table B-3 lists signal names and pin assignment.

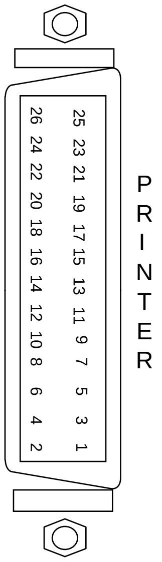


Figure B-3 Printer Cable Connector Pin Locations

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	$\overline{\text{DATA STROBE}}$	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	$\overline{\text{INPUT PRIME}}$
11	DATA6	24	GND
12	DATA7	25	Not connected
13	DATA8	26	Not connected

- 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.

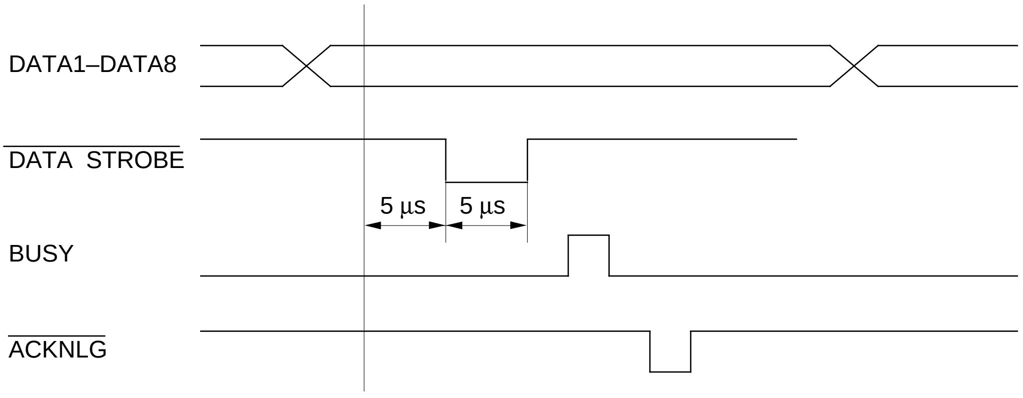
SELECT: If this signal is low, the E7000 sends no data and outputs the error message
***7: PRINTER NOT READY.

$\overline{\text{FAULT}}$: 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.



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.

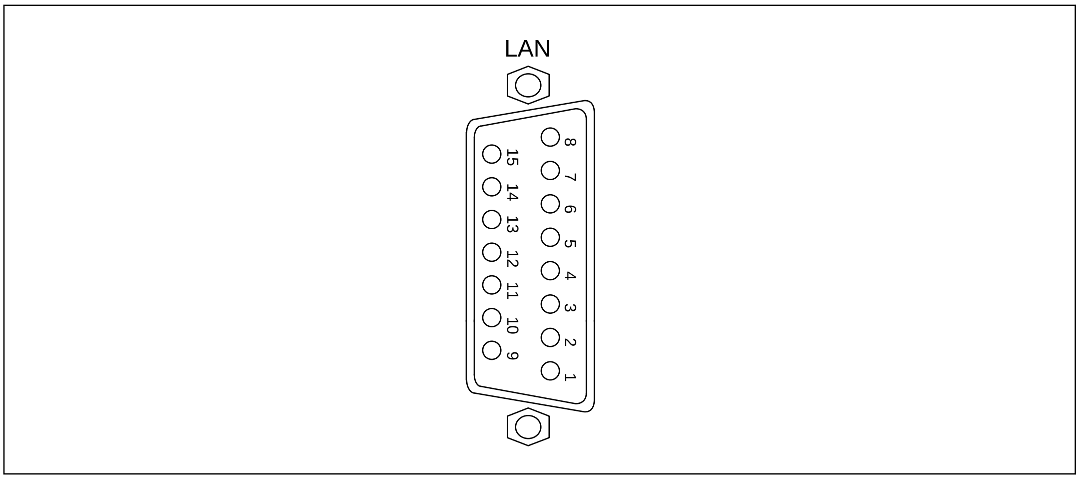


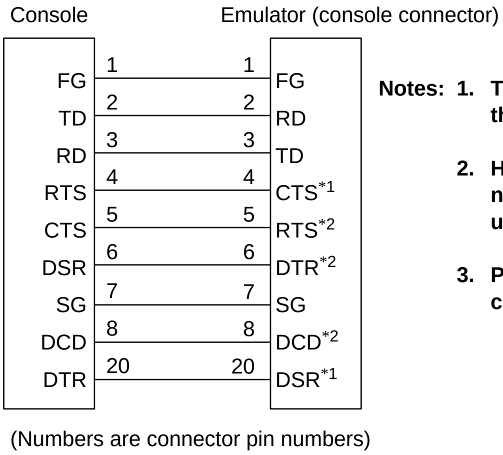
Figure B-5 LAN Connector Pin Locations

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

Pin	Signal Name
1	Not connected
2	COL+
3	TX+
4	—
5	RX+
6	GND
7	—
8	—
9	COL–
10	TX–
11	—
12	RX–
13	+12 V
14	—
15	—

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.

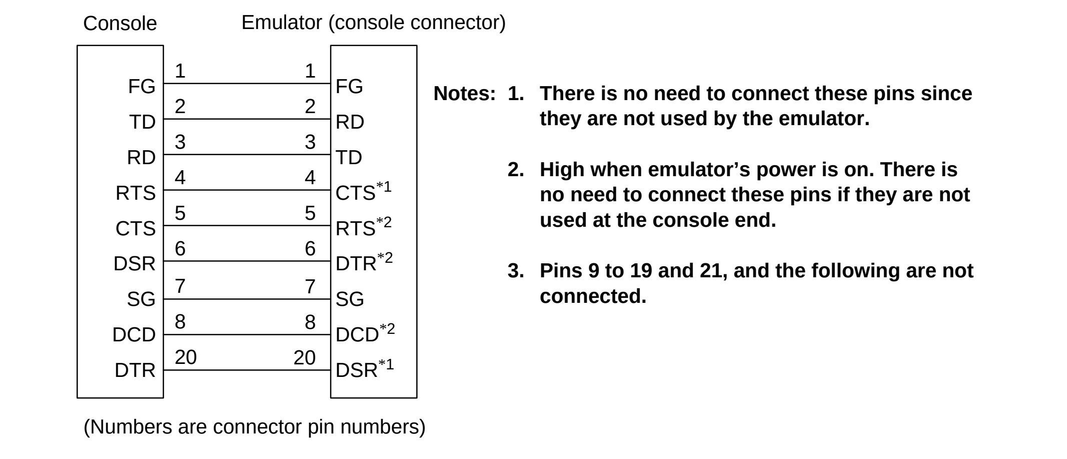
2. High when emulator’s power is on. There is no need to connect these pins if they are not used at the console end.

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

Figure B-6 Console to E7000 Wiring

B.6 E7000 to Host Computer Connection

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



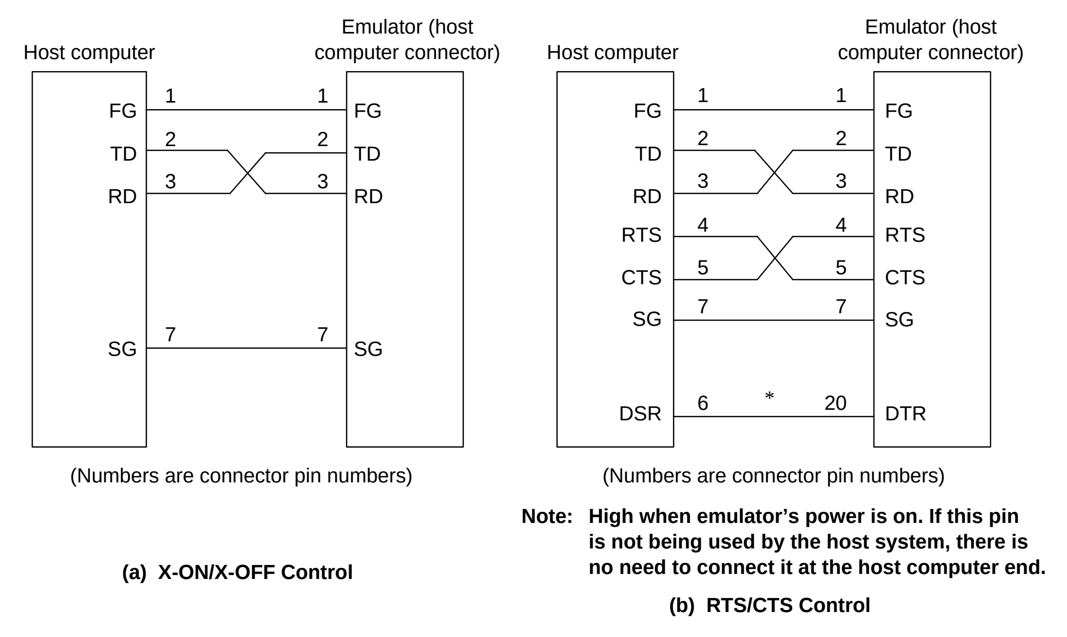
Notes: 1. There is no need to connect these pins since they are not used by the emulator.

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 to 19 and 21, and the following are not connected.

Figure B-7 Host Computer to E7000 Wiring

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



(a) X-ON/X-OFF Control

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 computer end.

(b) RTS/CTS Control

Figure B-8 Host Computer 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	$\overline{\text{DATA STROBE}}$	1	3	
	GND	19	4	
2	DATA1	2	5	
	GND	20	4	
3	DATA2	3	6	
	GND	21	16	
4	DATA3	4	7	
	GND	22	9	
5	DATA4	5	8	
	GND	23	9	
6	DATA5	6	10	
	GND	24	18	
7	DATA6	7	11	
	GND	25	20	
8	DATA7	8	12	
	GND	26	14	
9	DATA8	9	13	
	GND	27	14	
10	$\overline{\text{ACKNLG}}$	10	15	
	GND	28	16	
11	BUSY	11	17	
	GND	29	18	
12	PE	12	19	
	GND	29	20	
13	SELECT	13	21	
	GND	20	22	
14	$\overline{\text{INPUT PRIME}}$	31	23	
	GND	30	24	
15	$\overline{\text{FAULT}}$	32	1	
	±0 V	16	2	
16	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.
	FG			
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 H8S/2655-series or H8S/2245-series microcomputer (collectively referred to as MCU), a member of H8S-series microcomputers, developed by Hitachi, Ltd.

The H8S/2655 series contains the high-speed CPU, internal RAM and ROM, a timer unit, a watchdog timer, serial communication interface, DMAC, DTC, I/O ports, and A/D and D/A converters on a single chip.

The H8S/2245 series contains the high-speed CPU, internal RAM and ROM, a timer unit, a watchdog timer, serial communication interface, DTC, I/O ports, and A/D and D/A converters on a single chip.

When the E7000PC is connected to a user system, it operates in place of the 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, emulator pod, and user system interface cable, as shown in figure 1-1.

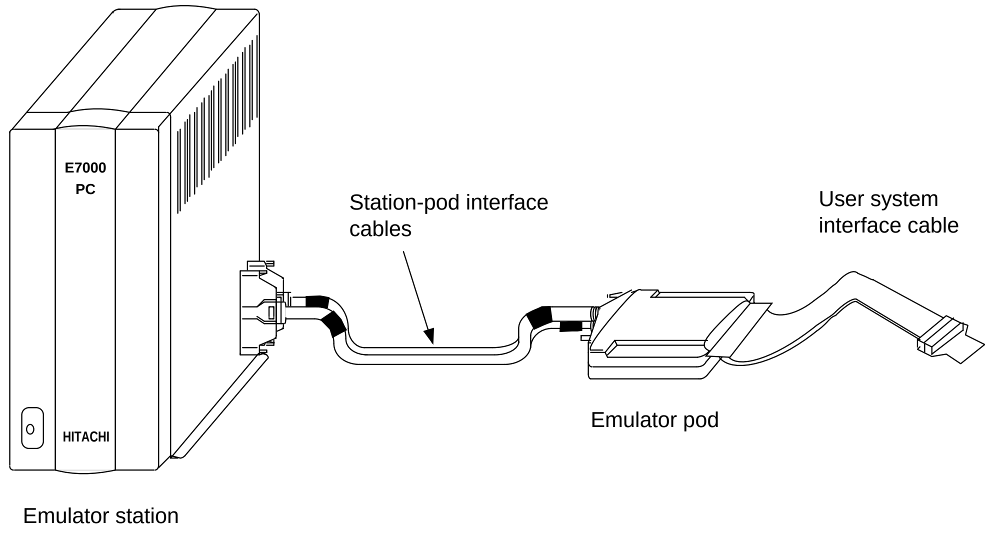


Figure 1-1 H8S/2655 E7000PC Emulator

The E7000PC provides the following features:

- Realtime emulation of the MCU
- 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 (option) 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 kbytes of emulation memory as substitute user system memory. An optional 1-Mbyte or 4-Mbyte emulation memory board can also be installed in the emulator station.
- By connecting the user system interface and providing a low-voltage power supply of 2.7 to 5.5 V, emulation can be performed in user systems with any supply voltage in the range 2.7 to 5.5 V.

1.1 Warnings

CAUTION

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

Before System Initiation:

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 at the rear panel.

7. After connecting a 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 MCU and the E7000PC, refer to section 2, Differences between the Actual MCU and the Emulator in Part III, Emulator Function Guide.

1.2 Environmental Conditions

CAUTION

If these instructions are not followed when using the emulator product, the user system and the emulator product will be damaged. The USER PROGRAM will be lost.

Observe the conditions listed in table 1-1 when using the E7000PC emulator.

Table 1-1 Environmental Conditions

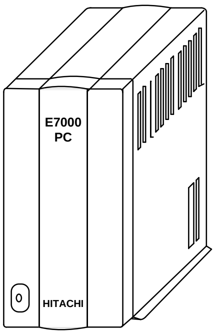
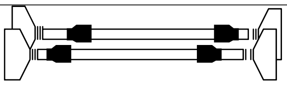
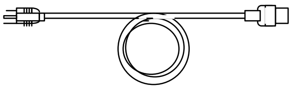
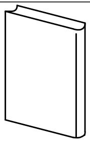
Item	**Specifications**
Temperature	Operating: +10 to +35°C Storage: –10 to +50°C
Humidity	Operating: 35 to 80% RH no condensation Storage: 35 to 80% RH no condensation
Vibration	Operating: 2.45 m/s^2 max. Storage: 4.9 m/s^2 max. Transportation: 14.7 m/s^2 max.
AC input power	Voltage: 100/200 VAC ±10% 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.

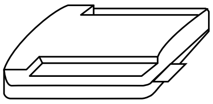
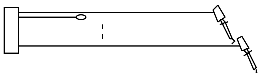
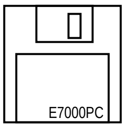
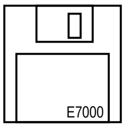
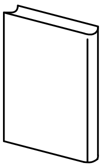
1.3.1 E7000PC Emulator Station

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
	External probe 	1	Signal input: Eight GND: One Trigger output: One
Software	Floppy disks 	1	E7000PC/IBM PC system program
		1	E7000 system program (cannot be used with E7000PC emulator)
Documentation	H8S/2655 Series E7000 Emulator User's Manual 	1	HS2655EPD70HE

Note: The E7000 system program cannot be used with the E7000PC emulator.

1.3.3 IBM PC Interface Board

Table 1-4 shows the specifications of the IBM PC interface board.

Table 1-4 IBM PC Interface Board

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	

1.3.4 Options

In addition to the emulator station and pod components, the options listed in table 1-5 are also available. Refer to each option manual for details on these optional components.

Table 1-5 Optional Component Specifications

Item	Model Name	Specifications
1-Mbyte emulation memory board	HS7000EMS11H	1-Mbyte SRAM is used
4-Mbyte emulation memory board	HS7000EMS12H	4-Mbyte SRAM is used
TQFP-120 user system interface cable	HS2655ECN71H	For H8S/2655 series (TFP-120)
QFP-128 user system interface cable	HS2655ECH71H	For H8S/2655 series (FP-128)
QFP-100 user system interface cable	HS2655ECH71H	For H8S/2245 series (FP-100B and TFP-100B)
Bus monitor interface board for E7000	HS7000EXR10H	For connecting the E7000 bus monitor board
E7000 bus monitor board	HS7000EBR01H	For installing the D/A converter

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.

CAUTION

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

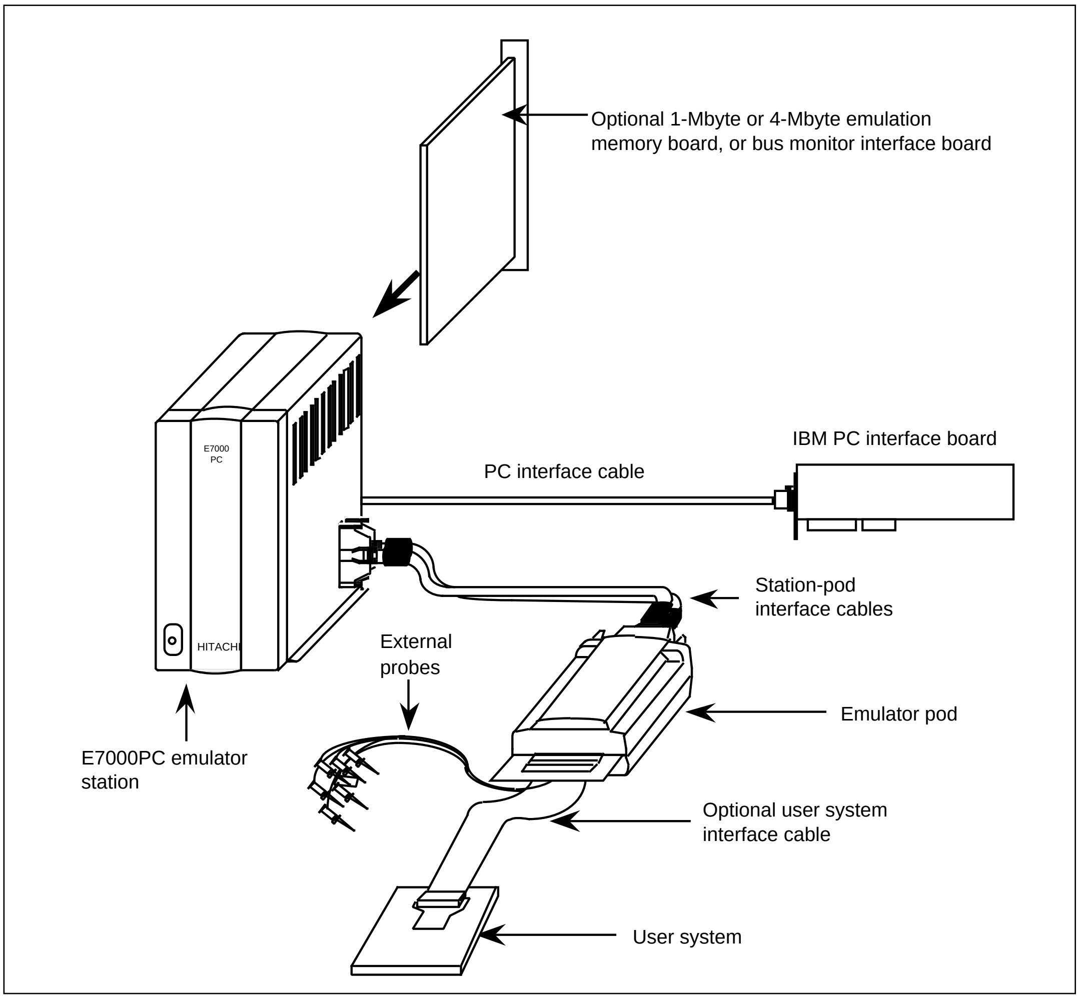

Optional 1-Mbyte or 4-Mbyte emulation memory board, or bus monitor interface board
IBM PC interface board
PC interface cable
E7000 PC
HITACHI
Station-pod interface cables
External probes
Emulator pod
E7000PC emulator station
Optional user system interface cable
User system

Figure 2-1 E7000PC Emulator Hardware Components

2.1.1 E7000PC Emulator Station Components

Front Panel:

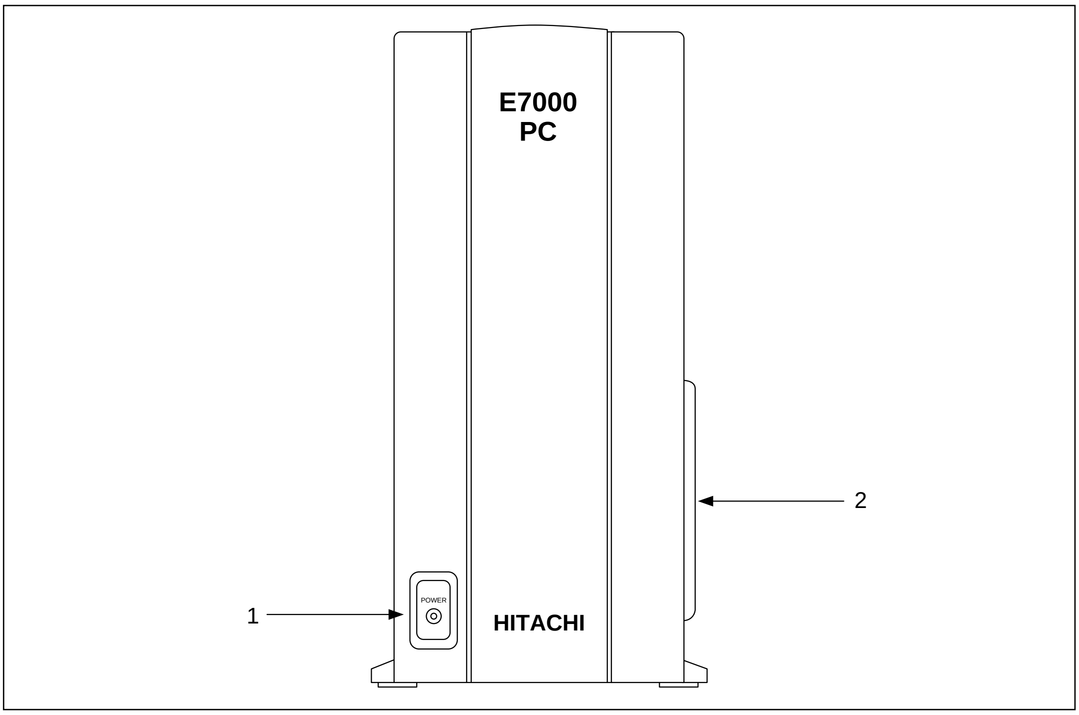


Figure 2-2 E7000PC Emulator Station Front Panel

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.

WARNING

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

Rear Panel:

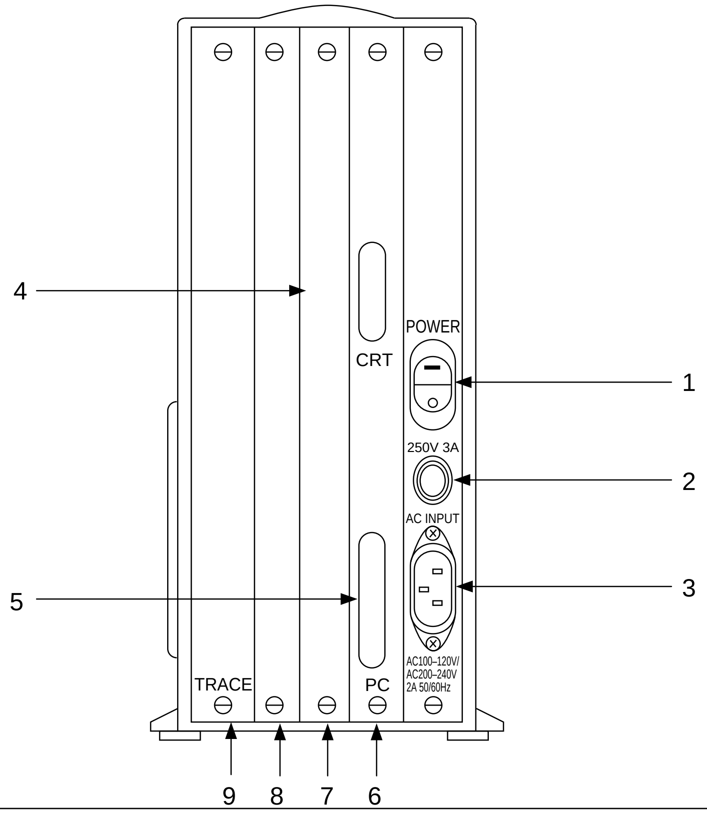


Figure 2-3 E7000PC Emulator Station Rear Panel

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

⚠ WARNING

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

2. Fuse box: Contains a 3-A 250 VAC fuse.

3. AC power connector: For a 100/200 VAC power supply.

4. Console connector For system extension use. Marked CRT.

5. Personal computer connector: For connection to the IBM PC console. Marked PC.

6. Control board slot: For installing the control board.

7. Extension slot: Cannot be used for the E7000PC.

8. Emulation memory /bus monitor interface board slot: For installing the optional emulation memory board or bus monitor interface board.

9. Trace board slot: For installing the trace board.

2.1.2 E7000PC Emulator Pod Components

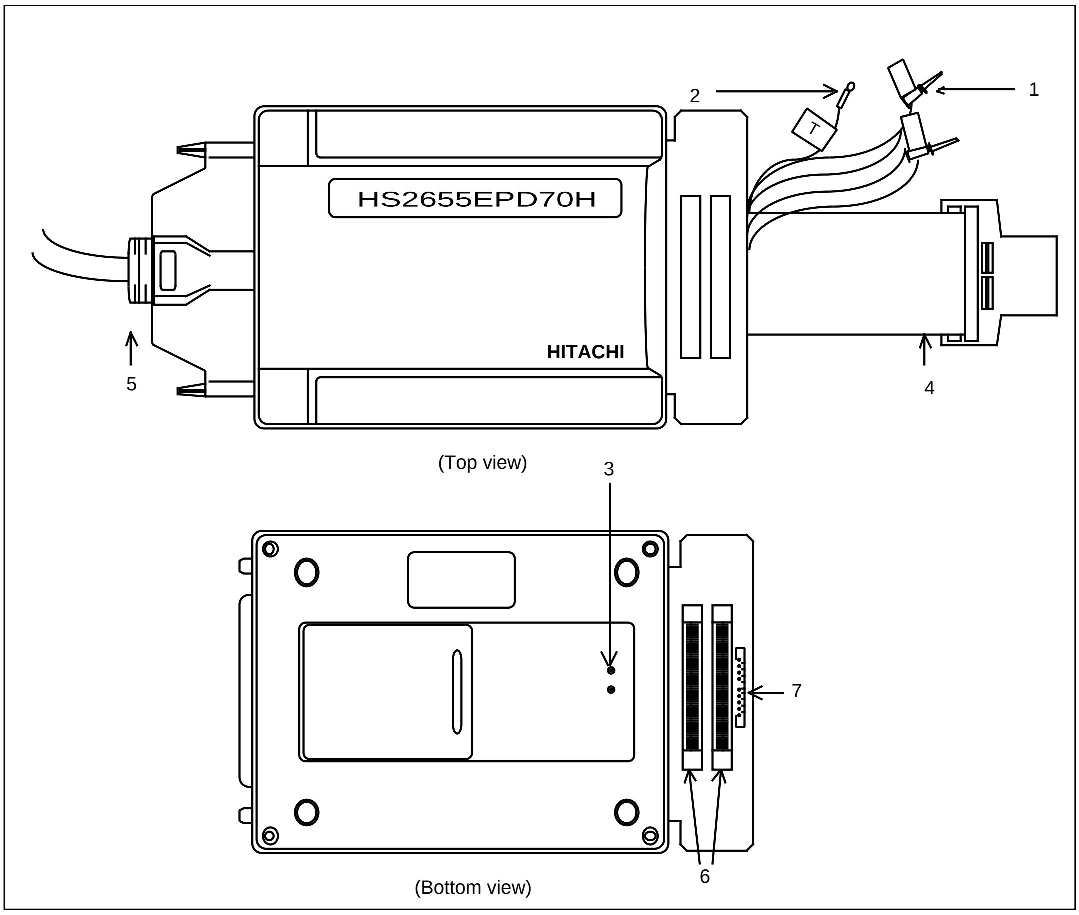


Figure 2-4 E7000PC Emulator Pod

1. External probes: Can be used for the following during user system emulation
   - — Hardware break condition input
   - — Realtime trace input
   - — Multibreak detection

2. Trigger output pin: Outputs a low-level pulse in the following states:
   - — When a hardware break condition is satisfied (whether to break or not can be selected)
   - — When cycle reset mode is specified with the GO command and an RES signal is input to the MCU
   - — When trigger output is specified with the TRACE_ CONDITION command and the specified conditions are satisfied, this output can be used as a trigger signal for an oscilloscope or a logic analyzer.

3. Crystal oscillator terminals: For installing a crystal oscillator to be used as a clock source for the MCU.

4. User system interface cable: For connection to the MCU socket on the user system, to enable the E7000 to operate in place of the MCU.

⚠ WARNING

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

CAUTION

When a user system interface cable is connected, power supply voltage must be provided from the VCC pin on the user system interface cable head to operate the E7000. Therefore, when operating the E7000 alone, be sure to disconnect the user system interface cable.

5. Station-pod interface cables: For connecting the emulator station to the emulator pod.

⚠ WARNING

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

6. User system interface cable connector: For connection of the user system.

⚠ WARNING

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

1. **Always switch OFF the emulator product and the user system before connecting or disconnecting the EMULATOR STATION CABLES or the EMULATOR POD.**

2. **When connecting the emulator pod to the user system, ensure that pin 1 of the user system connector on the emulator pod and that on the user system IC socket are correctly aligned.**

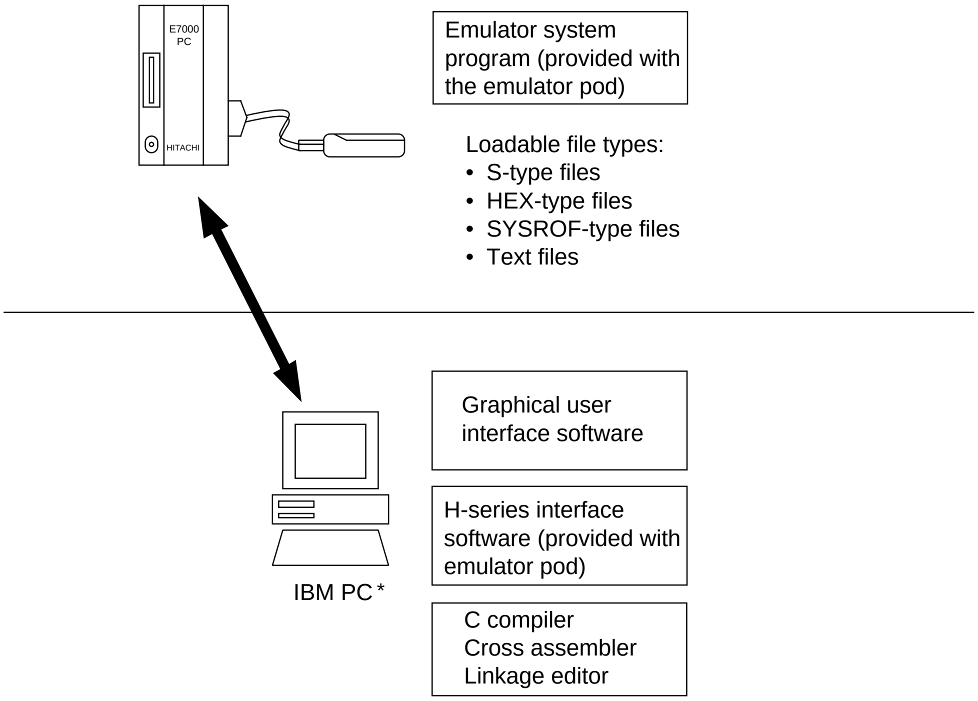
7. External probe connector: For connection of the external probe.

⚠ WARNING

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

2.2 E7000PC Software Components

The E7000PC emulator’s software components are illustrated in figure 2-5.


E7000
PC
HITACHI
Emulator system program (provided with the emulator pod)
Loadable file types:
• S-type files
• HEX-type files
• SYSROF-type files
• Text files
Graphical user interface software
H-series interface software (provided with emulator pod)
C compiler
Cross assembler
Linkage editor
IBM PC*

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

Figure 2-5 E7000PC Emulator Software Components

The emulator pod contains two 3.5-inch floppy disks; the E7000PC emulator system disk has “E7000PC” written under “HITACHI” on its label. Do not use the disk that has "E7000" on its label.

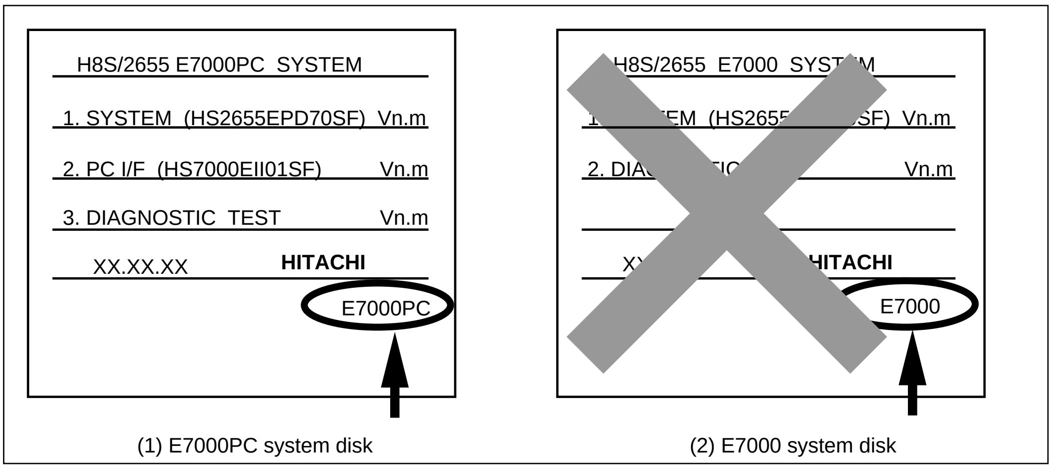


Figure 2-6 System Disk Labels

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 station memory after the E7000PC system program is activated.
H8S2655P.SYS	MCU control program	Controls the MCU within the emulator pod. Loaded into the emulator station memory after the E7000PC system program is activated.
H8S2655C.SYS	Configuration file	Contains MCU operating mode and MAP information. Loaded with the E7000PC system program.
LANCNF.SYS	LAN configuration file	Contains a host computer 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	Operates on an IBM PC to interface with the E7000PC.
DIAG.TM	Diagnostic program	Loaded into the emulator station memory for testing and maintenance.

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 2-7.

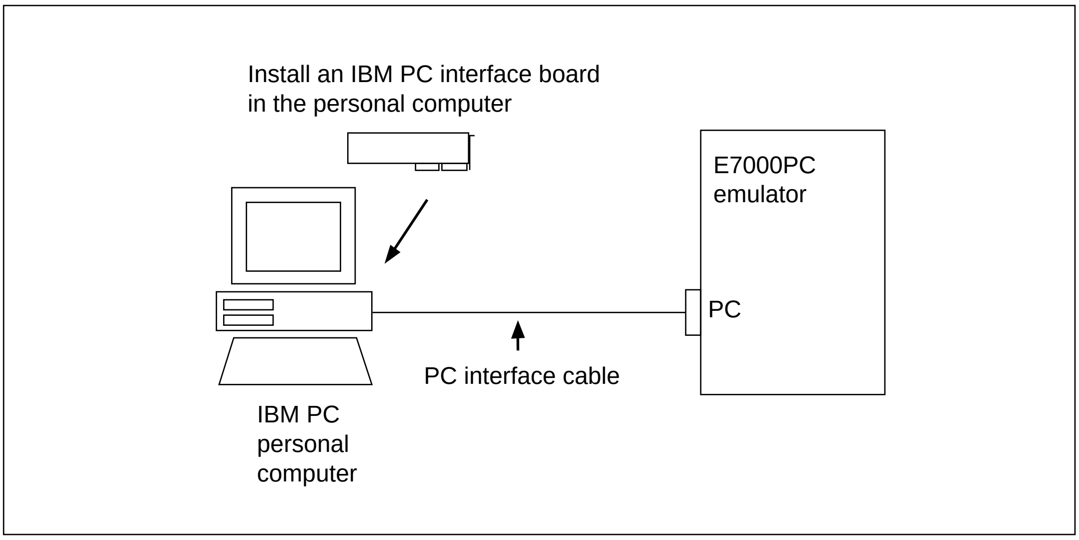


Figure 2-7 E7000PC Emulator System Configuration

Section 3   Preparation before Use

3.1 E7000PC Preparation

CAUTION

READ the reference sections shaded in figure 3-1 and the following warnings before using the emulator product. Incorrect operation will damage the user system and the emulator product. The USER PROGRAM will be LOST.

Unpack the E7000PC and prepare it for use as follows:

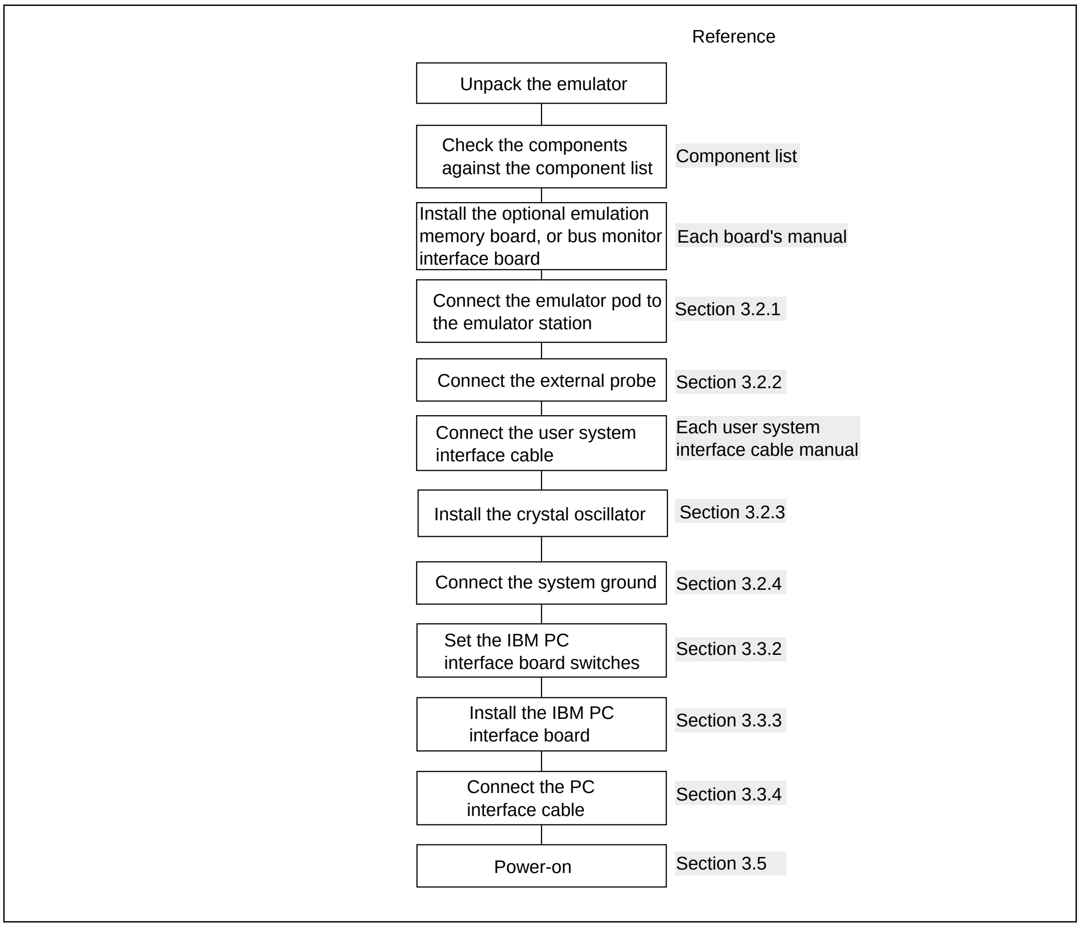

Reference
Unpack the emulator
Check the components against the component list
Component list
Install the optional emulation memory board, or bus monitor interface board
Each board's manual
Connect the emulator pod to the emulator station
Section 3.2.1
Connect the external probe
Section 3.2.2
Connect the user system interface cable
Each user system interface cable manual
Install the crystal oscillator
Section 3.2.3
Connect the system ground
Section 3.2.4
Set the IBM PC interface board 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
Section 3.5

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. Follow the procedure below to connect the emulator pod to the emulator station, or to disconnect it when moving the E7000PC:

⚠ WARNING

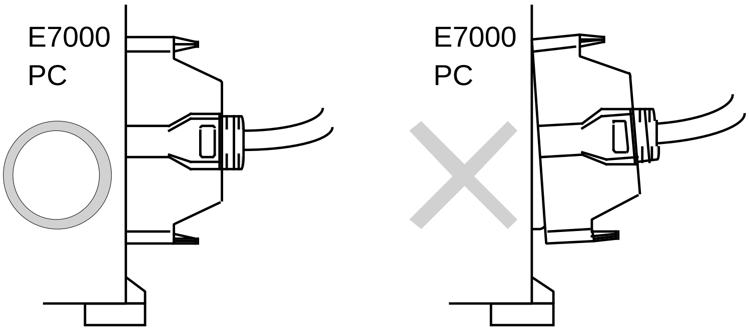
Always switch OFF the emulator product and the user system before connecting or disconnecting the EMULATOR STATION CABLES or the EMULATOR POD. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will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

(1) 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.

⚠ WARNING

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



(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.

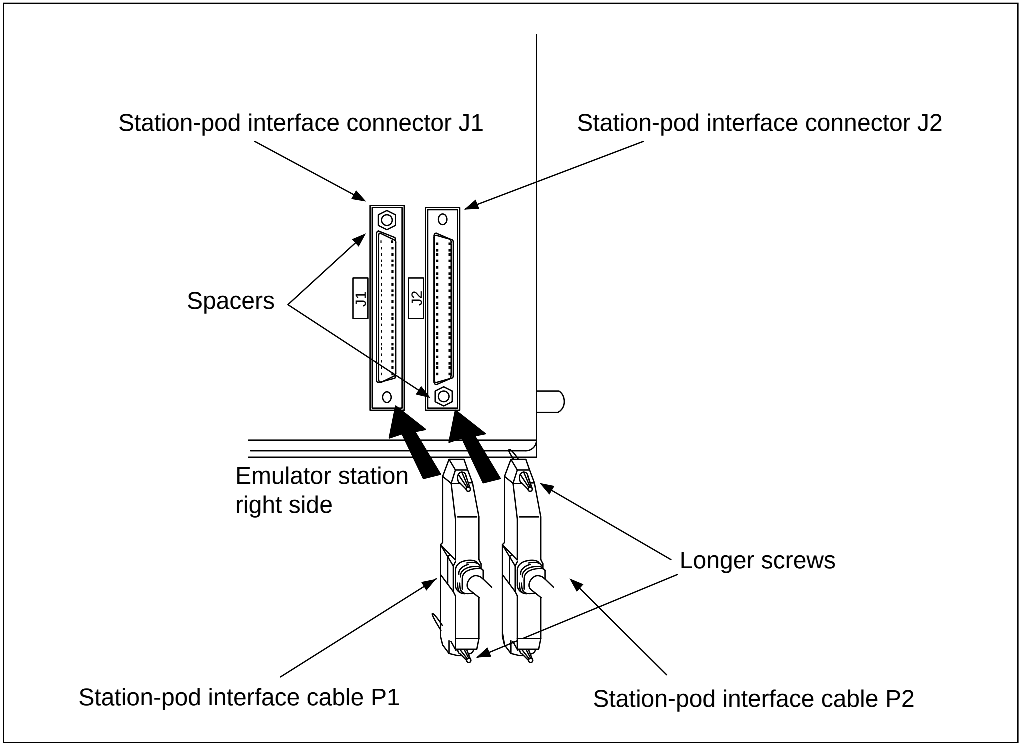

Station-pod interface connector J1
Station-pod interface connector J2
Spacers
J1
J2
Emulator station
right side
Longer screws
Station-pod interface cable P1
Station-pod interface cable P2

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 on the emulator pod, 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 3-3 shows how to connect the station-pod interface cables to the emulator pod.

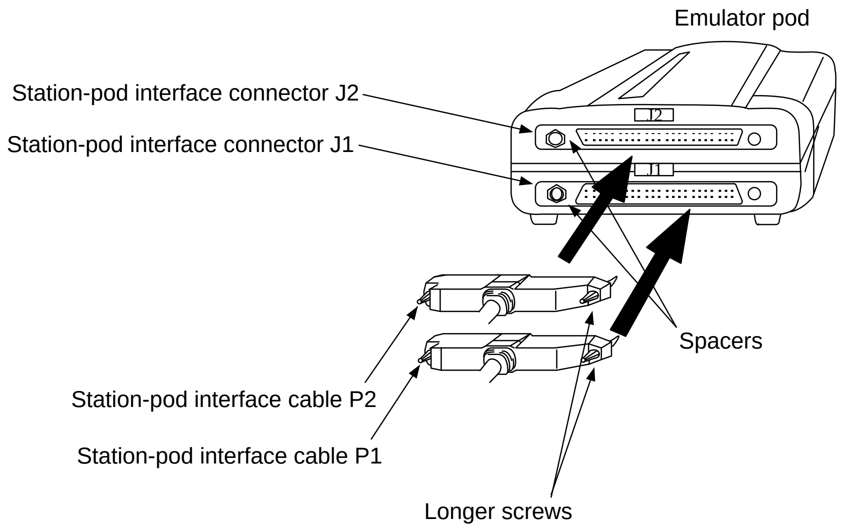

Emulator pod
Station-pod interface connector J2
Station-pod interface connector J1
J2
J1
Spacers
Station-pod interface cable P2
Station-pod interface cable P1
Longer screws

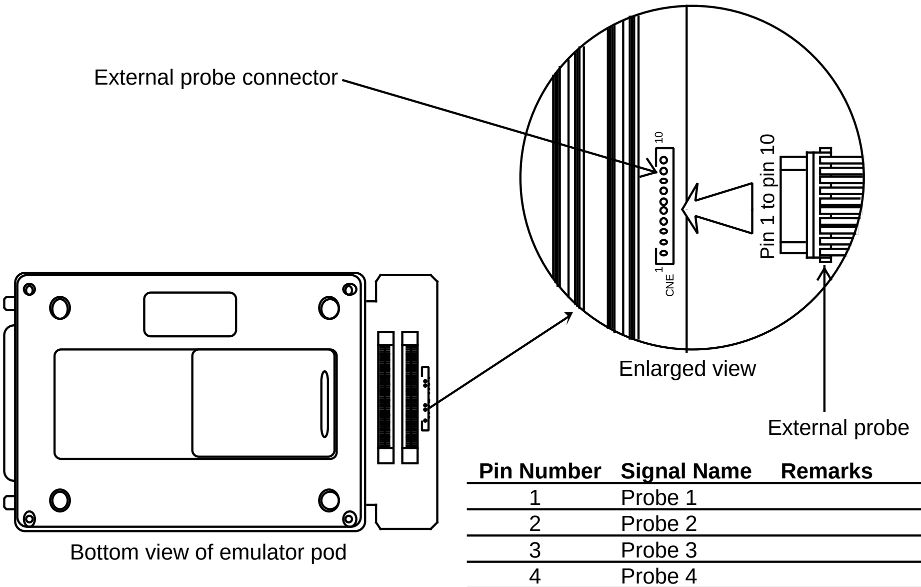
Figure 3-3 Connecting Station-Pod Interface Cables to Emulator Pod

3.2.2 External Probe Connector

CAUTION

Check the external probe direction and connect the external probe to the emulator pod correctly. Incorrect operation will damage the emulator pod and the external probe.

When an external probe is connected to the emulator pod, it enables external signal trace, break by external signal, and multibreak detection. Figure 3-4 shows the external probe connector. Check the external probe direction and connect the external probe to the emulator pod correctly.



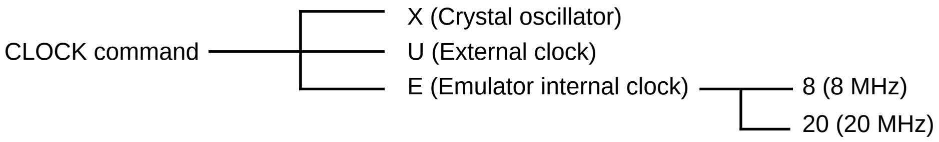
Pin Number	Signal Name	Remarks
1	Probe 1	
2	Probe 2	
3	Probe 3	
4	Probe 4	
5	Probe 5	
6	Probe 6	
7	Probe 7	
8	Probe 8	
9	GND	
10	Tirgger output	Output in trigger mode

Figure 3-4 Connecting External Probe

For connecting a user system interface cable, refer to the user system interface cable manual.

3.2.3 Selecting Clock

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



Crystal Oscillator: A crystal oscillator is not supplied with the emulator. Use one having the frequency used as the user system clock source (ϕ clock frequency), which must be from 8 MHz to 20 MHz. When using frequencies outside this range, supply an external clock from the user system.

WARNING

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

Follow the procedure below to install the crystal oscillator:

1. Check that the emulator power is off.

2. Install the crystal oscillator into the terminal at the bottom of the emulator pod (figure 3-5).

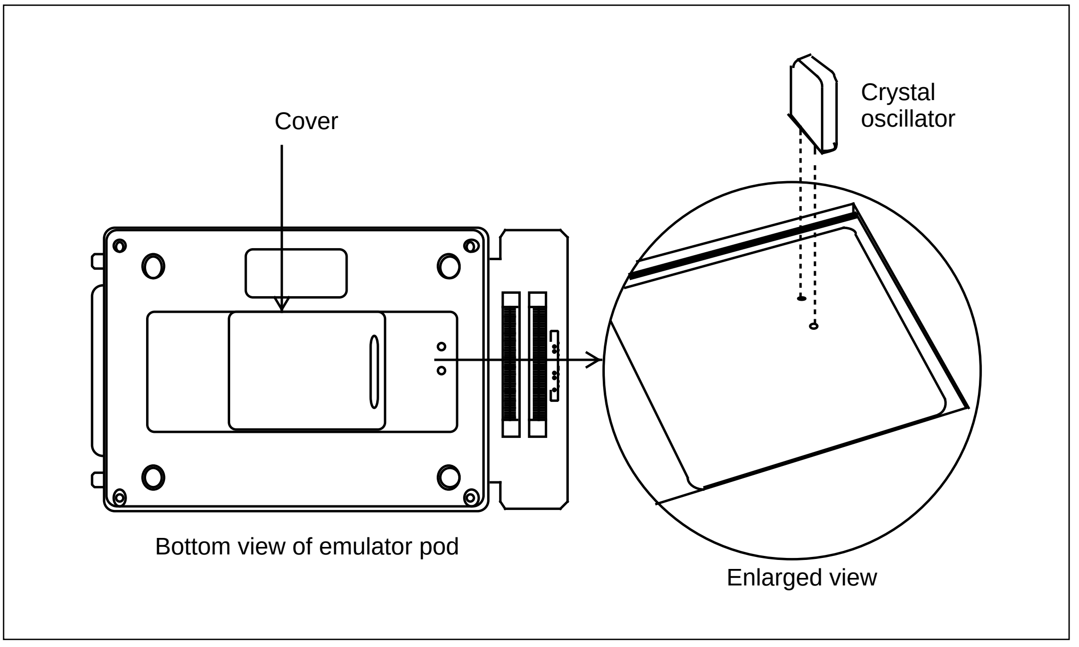


Figure 3-5 Installing the Crystal Oscillator

3. After turning on the emulator, specify X with the CLOCK command.

Using the crystal oscillator enables the user program to be executed at the user system operating frequency, even when the user system is not connected.

External Clock: Follow the procedure below to select the external clock.

⚠ WARNING

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

1. Check that the emulator power is off.

2. Connect the user system interface cable to the user system and supply a TTL-level clock through the EXTAL pin at the end of the user system interface cable from the user system.

3. After turning on the emulator, specify U with the CLOCK command.

Emulator Internal Clock: Specify 8 (8 MHz) or 20 (20 MHz) with the CLOCK command.

Reference: When the emulator system program is initiated, the emulator automatically selects the MCU clock source according to the following priority:
1. External clock (U) when supplied from the user system
2. Crystal oscillator (X) when installed in the emulator pod
3. 8-MHz emulator internal clock

3.2.4 Connecting System Ground

⚠ WARNING

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

The E7000PC 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 3-6). 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 power supply (figure 3-7).

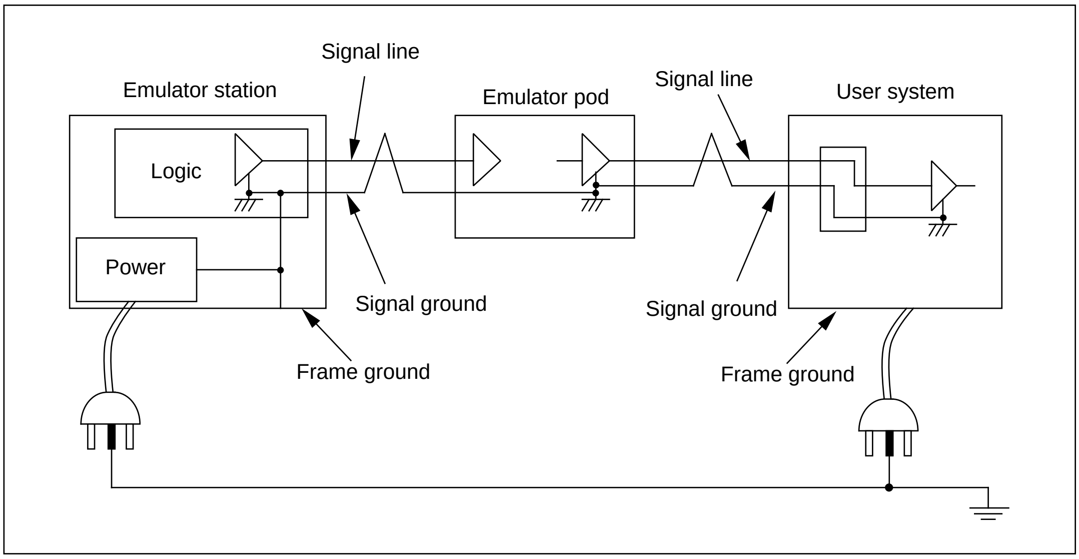


Figure 3-6 Connecting System Ground

⚠ WARNING

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

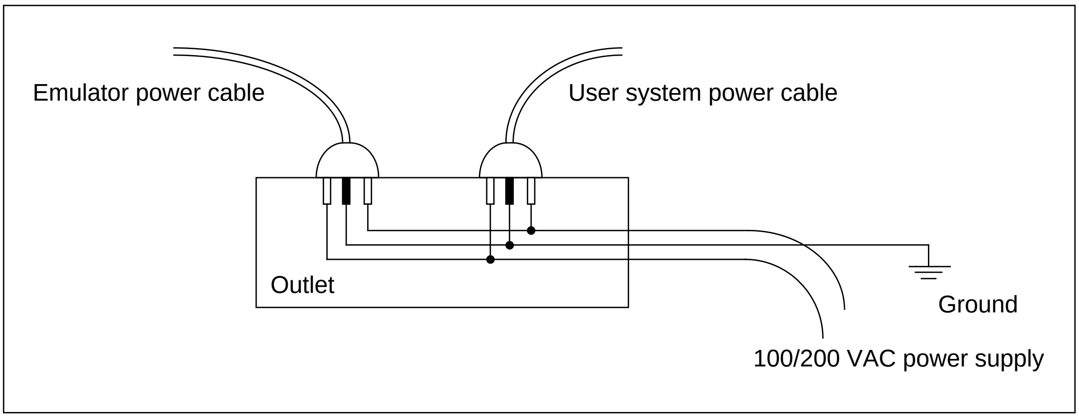


Figure 3-7 Connecting Frame Ground

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 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 the E7000PC GUI and IBM PC interface software that uses the IBM PC interface board.

3.3.2 Setting 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. It can be allocated to any 16-kbyte block within the address range from H'A0000 to H'FFFFF (figure 3-8). 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'D3FFF.

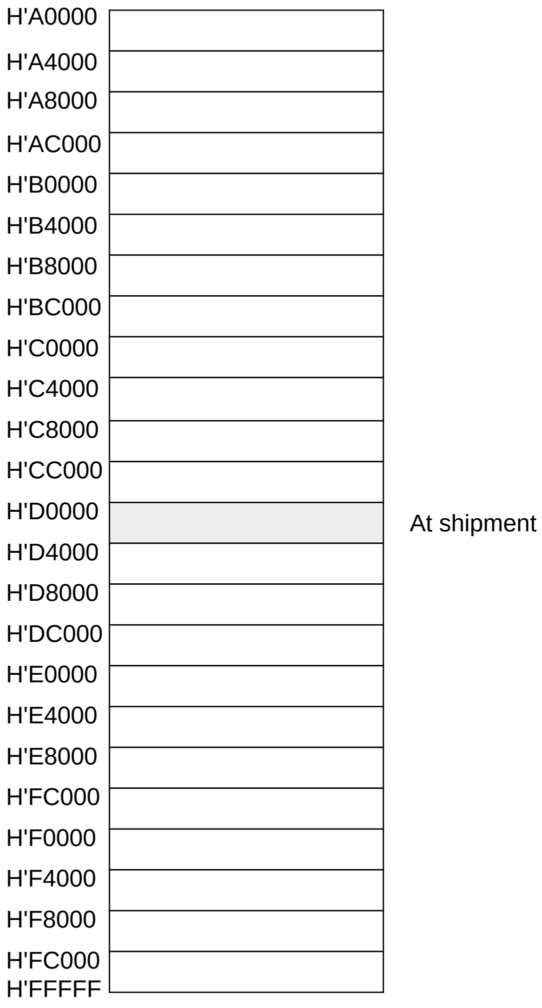


Figure 3-8 Memory Areas Allocatable for the IBM PC Interface Board

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 3-9 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.

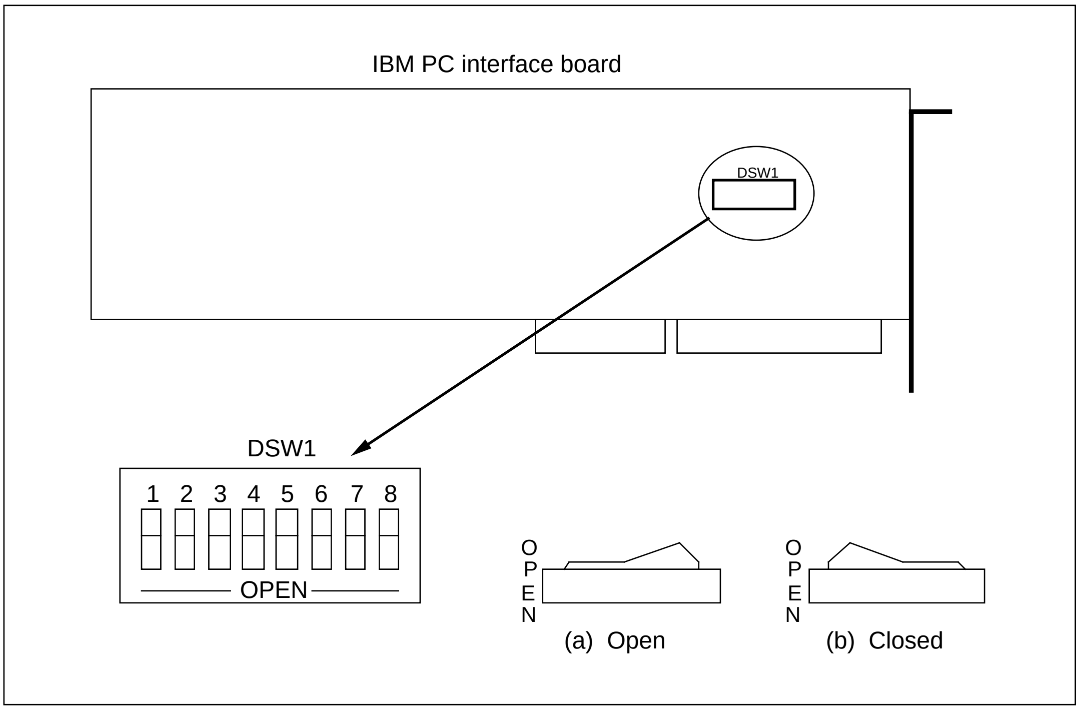


Figure 3-9 Switches on the IBM PC Interface Board

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

WARNING

Always switch OFF the emulator product, the user system, and the IBM PC before connecting or disconnecting the IBM PC interface board. Failure to do so will result in a FIRE HAZARD and will damage the user system, the IBM PC, and the emulator product or will result in PERSONAL INJURY.

Open the IBM PC cover and install the IBM PC interface board into an extension 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.

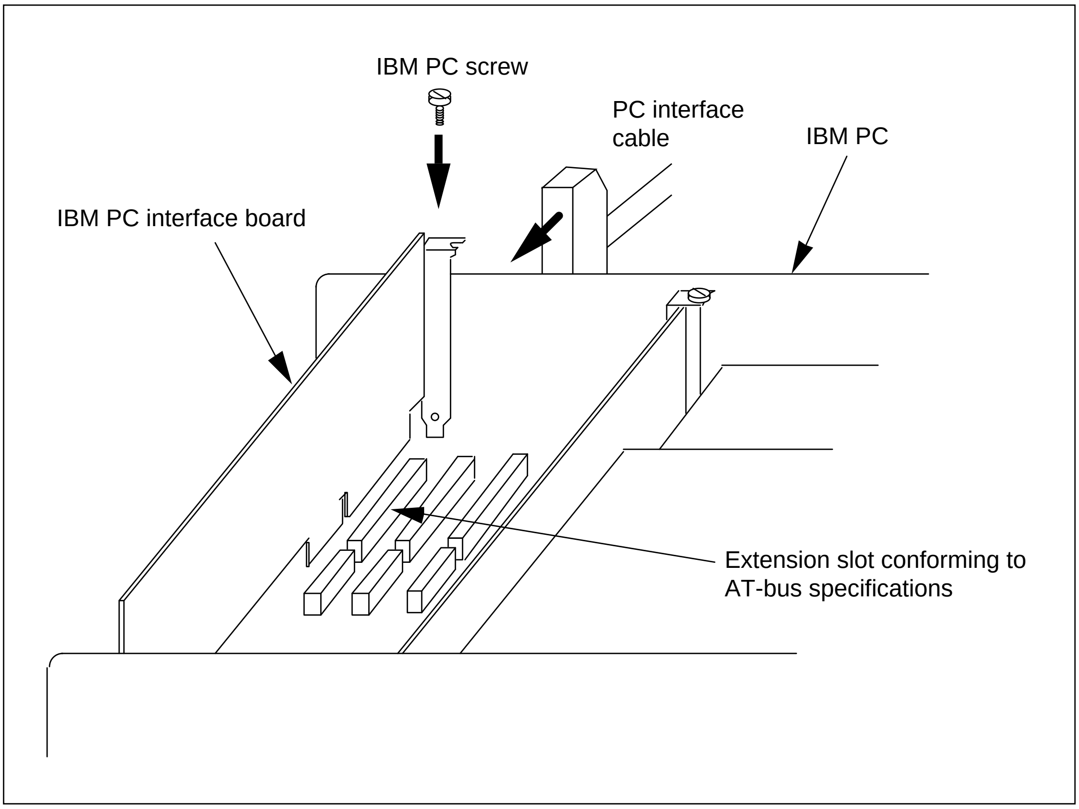


Figure 3-10 Installing IBM PC Interface Board

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

⚠ WARNING

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

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 3-11.

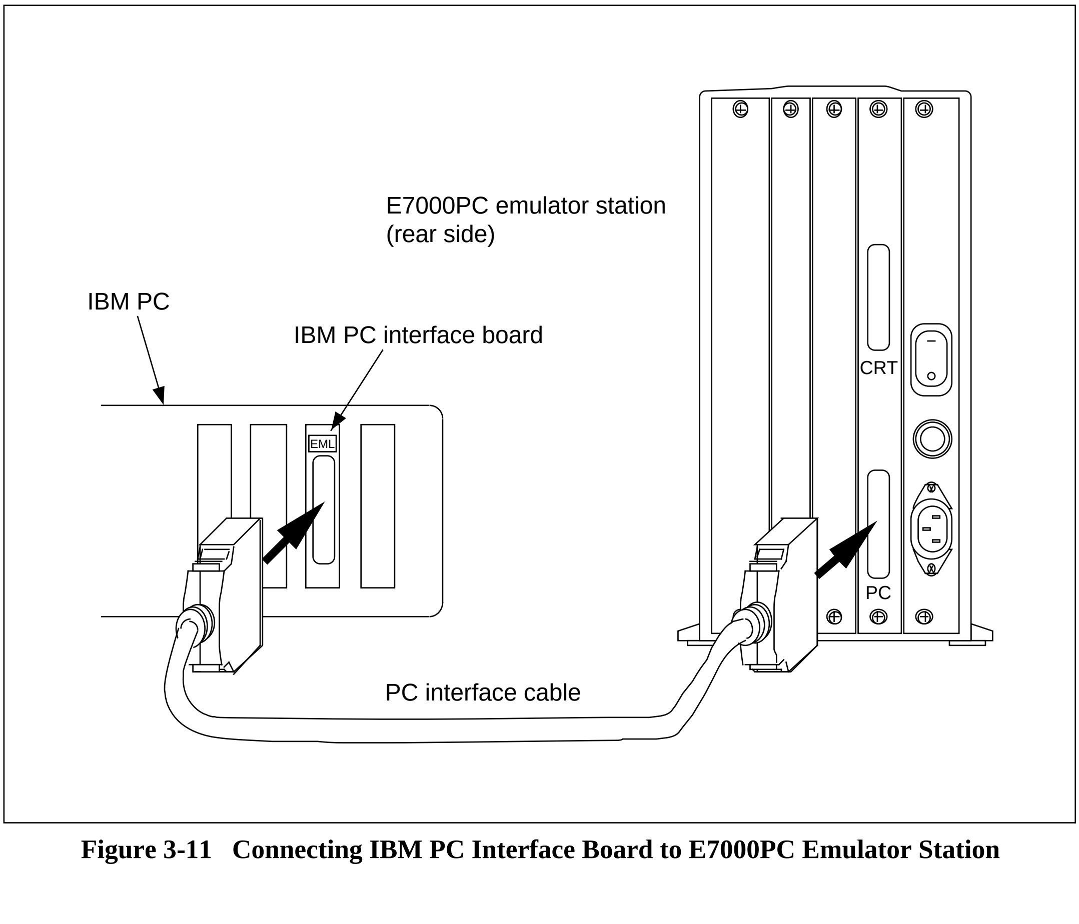


Figure 3-11 Connecting IBM PC Interface Board to E7000PC Emulator Station

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 H8S/2655 series 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).

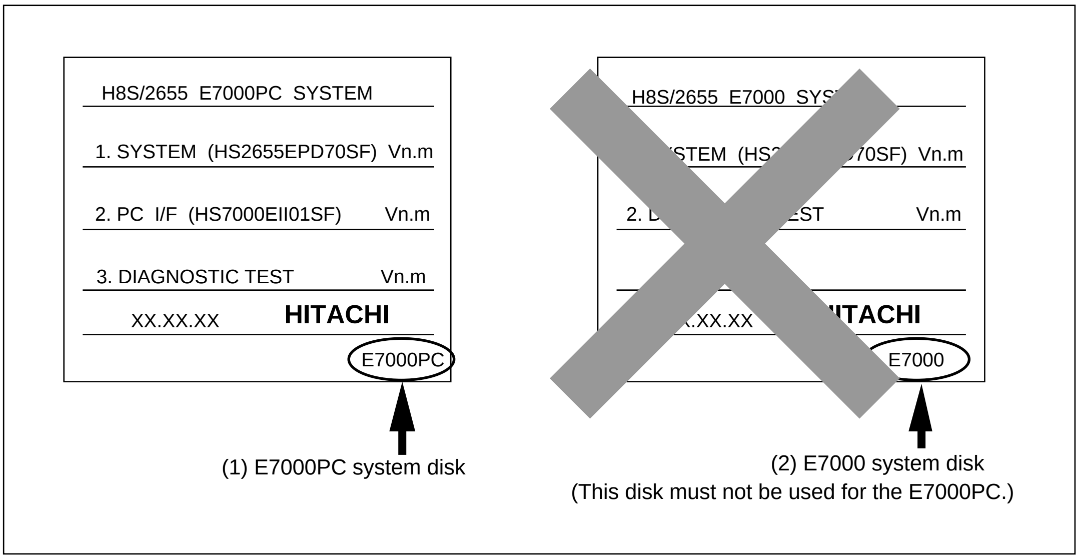


Figure 3-12 E70000PC System Disk

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

- E7000.SYS
- H8S2655P.SYS
- H8S2655C.SYS
- LANCNF.SYS (not used for the E7000PC)
- DIAG. TM
- IPI.EXE (IBM PC interface software)

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 H8S/2655 series emulator pod, or the H8S/2655 series 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*.

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

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*. 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\H8S2655, the termination code is 04 (CTRL+D), and the interface board is installed at the address D8000:0000:

```
C>SET E7000SYS=C:\E7000\H8S2655, 04, D8 (RET)
```

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

Example:

```
C:\>SET E7000SYS=C:\E7000,1B,D0 ...........Specify the default value
C:\>SET E7000=C:\E7000\H8S/2655,1B,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)
```

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

Modifying CONFIG.SYS File: If the virtual EMS driver (EMM386) 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 as E000:0000.

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

Example:

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

CAUTION

The CONFIG.SYS file should be backed up before modification.

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

Specify EMMExclude in section [386Enh] of the SYSTEM.INI file by entering the underlined section shown below. How to specify addresses and address ranges varies depending on the IBM PC system software used. For more details, refer to the Windows manual.

Example:

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

CAUTION

The CONFIG.SYS file should be backed up before modification.
Modifying the SYSTEM.INI file is not necessary
if the Windows is not used

Note: Windows is a trademark of Microsoft Corporation.

3.5 Power-On and Power-Off Procedures for the E7000PC

Figure 3-13 shows the E7000PC power-on procedure when the RS-232C interface is used.

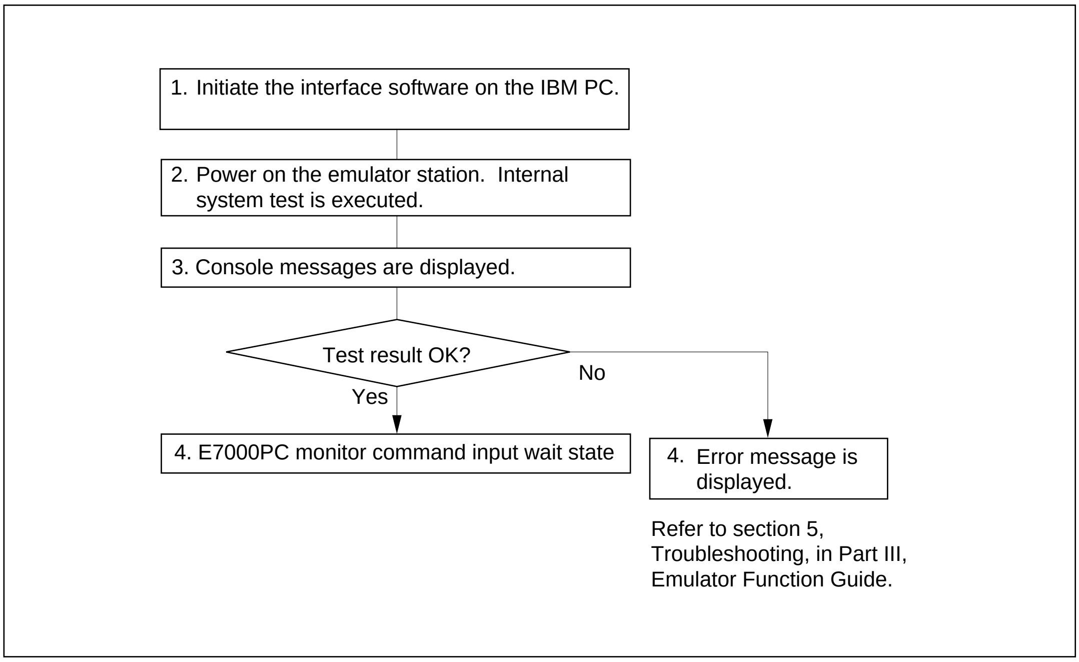

1. Initiate the interface software on the IBM PC.
2. Power on the emulator station. Internal system test is executed.
3. Console messages are displayed.
Test result OK?
No
Yes
4. E7000PC monitor command input wait state
4. Error message is displayed.
Refer to section 5, Troubleshooting, in Part III,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

Figure 3-13 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.

To use the current settings when turning on the E7000PC the next time, save the settings with the QUIT command before turning off the E7000PC. For details on the QUIT command, refer to section 7.2.34, QUIT, in Part III, Emulator Function Guide.

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┐                                                  │
                   └R┘(RET)                                             ┘

**  E7000 SYSTEM LOADING **                                             ] (b)

H8S/xxxx E7000 (HSxxxx EPDxxSF) Vn.m                                    ┐
Copyright (C) Hitachi, Ltd. 199x                                        │ (c)
Licensed Material of Hitachi, Ltd.                                      ┘

CONFIGURATION FILE LOADING                                              ] (d)
HARD WARE REGISTER READ/WRITE CHECK                                     ] (e)
POD SYSTEM LOADING                                                      ] (f)
EMULATOR POD TEST                                                       ] (g)
**  RESET IN BY E7000  !                                                ] (h)
CLOCK = xx MHz                                                          ] (i)
CPU MODE=H8S/xxxx  OPERATION MODE=x(MDx-x=x)  MODE SET=xxxx             ┐
PIN MODE=xxx/xxx  INTERNAL ROM SIZE=xxxxB  INTERNAL RAM SIZE=xxB        │
ADC MODE=xxxxxx  DMAC MODE=xxxxxx  REFRESH MODE=xxxxxx                  │ (j)
SCI CHANNEL=xCH  16BIT TIMER MODE=TPUxxx                                ┘
REMAINS EMULATION MEMORY       S=xxxxx /E=xxxxx                         ] (k)

WARM OR COLD START                                                      ┐
  file name : WARM START                                                │
  return     : COLD START                                               │ (l)
  (file name/return) ? ┌<file name> (RET)┐                              │
                       └(RET)            ┘                              ┘
:                                                                       ] (m)
```

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

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

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

(h) A RES signal has been input to the MCU.

(i) Specified clock. If the user system is ready, the user system's clock is used. If the user system's clock is not ready, the crystal oscillator in the emulator pod is used. If the crystal oscillator is not ready, the 8-MHz E7000PC internal clock is used.

(j) MCU type, MCU operating mode, and number of pins. They are previously set with the MODE command (saved in configuration file). For details, refer to section 7.2.29, MODE, in Part III, Emulator Function Guide.

(k) Remaining emulation memory size

(l) 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.

(m) 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 7.2.34 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**
- **Performance analysis 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(RET)                                                   (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 of, zzzz indicates the offset of the memory where the interface board is installed. tt indicates the interface software termination 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) when the E7000PC is in the command input wait state, 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 1FFFF;U
 REMAINS EMULATION MEMORY S=xxxxx/E=xxxxx
: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 through the keyboard.

```
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 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 file type “$$$”, and will only be renamed with the specified file type 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 on the console.

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
000100    7A0700FF                 MOV.L        #00FFF000:32,ER7
          F000
000106    F800                     MOV.B        #00:8,R0L
000108    F900                     MOV.B        #00:8,R1L
00010A    8802                     ADD.B        #02:8,R0L
00010C    8901                     ADD.B        #01:8,R1L
00010E    A90A                     CMP.B        #0A:8,R1L
000110    46F8                     BNE          00010A:8
000112    6A881000                 MOV.B        R0L,@1000:16
000116    40FE                     BRA          000116:8
```

These examples assume that the emulator station is connected to the host system (IBM PC) via the PC interface, that the E7000PC system program has been installed in the host system, 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:

Note: Only parts of the console message are shown depending on the amount of vacant space in this document.

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. The console displays the message shown on the right when the PC interface program operates.
   If this message is not displayed, follow the troubleshooting procedure in section 5, Troubleshooting, in Part III, Emulator Function Guide.

```
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
```

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

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

4.1 Basic Examples

4.1.1 Preparing for Connection of IBM PC

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

Operations **Console Message**

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.

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

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

```
** E7000 SYSTEM LOADING **

H8S/xxxx E7000 (HSxxxxEPDxxSF) Vn. m
Copyright (C) Hitachi, Ltd. 199x
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING
HARD WARE REGISTER READ/WRITE CHECK
POD SYSTEM LOADING
EMULATOR POD TEST
** RESET IN BY E7000 !
CLOCK = xx MHz
```

```
CPU MODE = H8S/xxxx  OPERATION MODE=x(MD x-x
PIN MODE=xxx/xxx  INTERNAL ROM SIZE=xxxxB  IN
ADC MODE=xxxxxx  DMAC MODE=xxxxxx  REFRESH
SCI CHANEL=xCH  16BIT  TIMER MODE=TPUx-x
REMAINS EMULATION MEMORY   S=xxxxx/E=xxxxxx

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

3. Enter (RET).

4.1.2 Specifying the MCU Operating Mode

Specify the MCU operating mode by the following procedure.

⚠ WARNING

Correctly specify the operating mode according to the TARGET DEVICE settings (table 4-1). The emulator will not operate correctly with an incorrect setting. Failure to do so will result in a FIRE HAZARD and will damage the user system and the emulator product or will result in PERSONAL INJURY. The USER PROGRAM will be LOST.

In the following example, MCU operating mode 4 of the H8S/2653 is selected. When using other target devices, specify the operating mode according to the target device settings shown in table 4-1.

Table 4-1 Target Device Settings

	Target Device		
Item	**H8S/2653**	**H8S/2655**	**H8S/2245**
CPU MODE	2	2	1
OPERATION MODE	1 to 7	1 to 7	1 to 7
MODE SET	1 or 2	1 or 2	1 or 2
PIN MODE	4	4	6
INTERNAL ROM SIZE	3 (64 kbytes)	5 (128 kbytes)	5 (128 kbytes)
INTERNAL RAM SIZE	4 (4 kbytes)	4 (4 kbytes)	4 (4 kbytes)
ADC MODE	2	2	1
DMAC MODE	2	2	1
REFRESH MODE	2	2	1
SCI CHANNEL	2	2	2
16BIT TIMER MODE	2	2	1

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 the H8S/2653 as the target MCU, for example, enter 2 (RET).

```
CPU MODE (1:H8S/2000, 2:H8S/2600) x  ?  _

CPU MODE (1:H8S/2000, 2:H8S/2600) x  ?  2 (RET)
```

3. To select mode 4 as the MCU operating mode, for example, enter 4 (RET).

```
OPERATION MODE (MD2-0) x  ?  4  (RET)
```

4. The method for setting the operating mode is specified as follows:
   1: Selects the operating mode set in step 2 above.
   2: Selects the operating mode set by the operating mode selection pin (MD2 to MD0) on the user system

   To use the emulator without connecting the user system, for example, enter 1 (RET).

```
MODE SET (1:E7000 MODE, 2:USER MODE) x ?  1 (RET)
```

5. To select the TFP-120 package as the number of pins, for example, enter 4 (RET).

```
PIN MODE (1:80/84-A, 2:100-A, 3:112, 4:120/128, 5:80/84-B, 6:100-B) x ? 4 (RET)
```

6. To select 64 kbytes as the internal ROM size because the H8S/2653 has 64-kbyte ROM, enter 3 (RET).

```
INTERNAL ROM SIZE (1:0KB, 2:32KB, 3:64KB, 4:96KB, 5:128KB, 6:256KB) x ? 3 (RET)
```

7. To select 4 kbytes as the internal RAM size because the H8S/2653 has 4-kbyte RAM, enter 4 (RET).

```
INTERNAL RAM SIZE (1:1KB, 2:2KB, 3:3KB, 4:4KB, 5:6KB, 6:8KB) x ?  4 (RET)
```

8. Select the internal AD converter.

```
ADC MODE (1:NORMAL, 2:HI SPEED) x ? 2 (RET)
```

9. Select the internal DMAC.

```
DMAC MODE (1:DISABLE, 2:ENABLE) x ? 2 (RET)
```

10. Select whether to enable the internal refresh controller.

```
REFRESH MODE (1:DISABLE, 2:ENABLE) x ? 2 (RET)
```

11. Select the number of SCI channels.

```
SCI CHANEL (1:2CH, 2:3CH) x ? 2 (RET)
```

12. Select a combination of 16-bit timers.

```
16BIT TIMER MODE (1:TPU0-2, 2:TPU0-5) x ? 2 (RET)
```

13. 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 2 above before entering Y (RET) and repeat the procedure.

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

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

14. After the mode settings have been stored in the configuration file, the E7000PC system program automatically terminates. Re-initiate the E7000PC system.

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

4.1.3 Allocating Standard Emulation Memory and Specifying Attributes

In order to load the user program to memory and execute it, allocate the standard emulation memory in the emulator pod by the following procedure:

Operations

Enter MAP 0 1FFFF;S (RET) to allocate the standard emulation memory to addresses H'0 to H'1FFFF. The console displays the message shown on the right, which indicates that the memory allocation has been completed.
Enter MAP (RET) and the console displays the attributes of all the memory areas.

Console Message

```
:MAP 0 1FFFF;S (RET)


 REMAINS EMULATION MEMORY     S=60000/E=000000


:MAP (RET)
 00000000-0001FFFF;S   00020000-00FFFFFF;U
 INTERNAL RAM  =  00FFEC00-00FFFBFF
 INTERNAL I/O      =  00FFFE40-00FFFFFF
 REMAINS EMULATION MEMORY     S=60000/E=000000
:
```

4.1.4 Executing Program

Execute the loaded program by the following procedure:

Operations **Console Message**

1. Set the initial values of the registers. Enter .SP (RET) then FFF000 (RET) as the SP value to set the stack pointer (SP register) to H'FFF000. The console then asks for the program counter value. Enter 100 (RET) as the program counter value. The console then asks for the condition code register value. In this example, other registers need not be set or changed, therefore, enter . (RET) to exit this interactive mode.

```
:.SP (RET)
 ER7(SP) = 00FFF7FE ? _

 ER7(SP) = 00FFF7FE ? FFF000 (RET)
 PC         = 00FFFFFF ? _

 PC         = 00FFFFFF ? 100 (RET)
 CCR     = 80:I****** ? _

 CCR     = 80:I****** ?  . (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 .(RET) to the condition code register prompt can complete the register modification procedure. A register value can also be directly input without using the interactive mode. For example, to set the stack pointer value directly, enter .SP FFF000 (RET).

2. Enter GO (RET) to execute the loaded program from the address pointed to 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 registers such as the program counter, the condition code register, and the general registers ER0 to ER7 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 keys were entered.

```
:(BREAK)

 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:01S:049705US
 +++: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 10C (RET) to terminate program execution immediately before the instruction at address H'10C in the program has been executed. The instruction at the address where a software breakpoint is set is replaced with a break instruction.

```
:BREAK 10C (RET)
```

2. Restart program execution from address H'100. 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'100, then enter GO, as described in section 4.1.5, Executing Program.

```
:GO 100 (RET)
 **  PC = xxxxxxxx
```

3. The program execution terminates immediately before the instruction at address H'10C has been executed. The console displays the data shown on the right. BREAK POINT 0000010C shows that the program execution was terminated because of a software breakpoint at address H'10C.

```
 PC =0000010C   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000002  00000000  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:00S:000007US
 +++:BREAK POINT  0000010C
:_
```

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. 0000010C ADD.B #01:8,R1L shows the executed address and mnemonic code, and +++:STEP NORMAL END shows that the single-step execution has terminated.

```
:STEP (RET)



 PC =0000010E   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000002  00000001  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 0000010C                  ADD.B         #01:8,R1L
 +++:STEP NORMAL END
:_
```

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

```
:(RET)
 PC =00000110   CCR =A9:I*H*N**C   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000002  00000001  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 0000010E                  CMP.B         #0A:8,R1L
 +++: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'1000, enter BREAK_CONDITION1 A = 1000 W (RET).

```
:BREAK_CONDITION1 A = 1000 W (RET)
```

4. Enter GO 100 (RET) to start executing the program from address H'100. 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 100 (RET)
 **  RUNNING
 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D'0000H:00M:00S:000018US
 +++: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 instruction mnemonic information of the executed instructions.

```
:TRACE (RET)
```

```
     IP         ADDR       LABEL   MNEMONIC    OPERAND
*-D'00044     00000100             MOV.L       #00FFF000:32,ER7
*-D'00043     00000106             MOV.B       #00:8,R0L
*-D'00042     00000108             MOV.B       #00:8,R1L
*-D'00041     0000010A             ADD.B       #02:8,R0L
*-D'00040     0000010C             ADD.B       #01:8,R1L
    :            :                    :            :
```

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

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

```
     BP        AB      DB    MA   R/W  ST    IRQ      NMI  RA   PROG       CLK
*          00000100               MOV.L    #00FFF000:32,ER7
 -D'00062  00000100  7A07  EXT   R   PRG  11101111  1   11   11111111   07
 -D'00061  00000102  00FF  EXT   R   PRG  11101111  1   11   11111111   06
 -D'00060  00000104  F000  EXT   R   PRG  11101111  1   11   11111111   06
*          00000106               MOV.B    #00:8,R0L
  -D'00059 00000106  F800  EXT   R   PRG  11101111  1   11   11111111   06
*          00000108               MOV.B    #00:8,R1L
  -D'00058 00000108  F900  EXT   R   PRG  11101111  1   11   11111111   06
                :                            :                  :
```

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 10A 5 (RET) to terminate program execution immediately after address H'10A is passed five times.

```
:BREAK  10A  5 (RET)
```

2. To start execution from address H'100, enter GO 100 (RET).

3. When execution terminates after address H'10A is passed five times, the console displays the data shown on the right.

```
:GO 100 (RET)
 **  RUNNING
 PC =0000010A   CCR =A9:I*H*N**C  EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000008  00000004  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D'0000H:00M:00S:000040US
 +++:BREAK POINT 0000010A
:_
```

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>
 0000010A          0005       0005
    (a)                  (b)           (c)
```

4.2.2 Conditional Trace

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

Operations **Console Message**

1. Enter BREAK - (RET) and BREAK_CONDITION1 - (RET) to cancel the breakpoints set in the example of section 4.2.1, Break with Pass Count Condition, and section 4.1.7, Setting Hardware Break Conditions.

```
:BREAK - (RET)
:BREAK_CONDITION1 - (RET)
```

2. Enter TRACE_CONDITION A =100:106;R (RET) to get trace information only while the program counter is between addresses H'100 and H'106.

```
:TRACE_CONDITION A= 100:106;R (RET)
```

3. Enter GO 100 (RET) to start executing the program, then (BREAK) key or (CTRL) + C keys to terminate the execution.

```
:GO 100 (RET)
 **  PC = xxxxxxxx  (BREAK)
 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:01S:430529US
 +++:BREAK KEY
:_
```

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

```
:TRACE (RET)

     IP          ADDR       LABEL    MNEMONIC     OPERAND
  *-D'*****     00000100             MOV.L        #00FFF000:32,ER7
  *-D'*****     00000106             MOV.B        #00:8,R0L
      :            :                    :            :
```

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 100 (RET)
 ** PC = xxxxxxxx     (RET)
#_
```

2. Enter DUMP 1000 100F (RET) to display the memory contents from H'1000 to H'100F in parallel mode.

```
#DUMP 1000 100F (RET)
```

```
<ADDR>                    <     D    A       T     A      >                      <ASCII CODE>
00001000   14  00  00  00  00  00  00  00      00  00  00  00  00  00  00  00       “. . . . . . . .”
```

3. Enter MEMORY 117 EE (RET) to modify the contents of memory address H'117 into EE in parallel mode.

```
#MEMORY 117 EE (RET)
```

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

```
#END (RET)
 ***81:  TRACE CONDITION RESET
 ** PC = xxxxxxxx
```

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

```
(BREAK)
 PC =00000116   CCR =80:I*******   EXR=07:*****210
 MACH=00000000   MACL=00000000
 ER0 - ER3   00000014  0000000A  00000000  00000000
 ER4 - ER7   00000000  00000000   00000000  00FFF000
 RUN - TIME = D’0000H:00M:00S:0430529US
 +++:BREAK KEY
:_
```

6. Enter DISASSEMBLE 100 117 (RET) to confirm that the program has been changed by memory modification in parallel mode.

```
:DISASSEMBLE 100 117 (RET)
```

```
ADDR          CODE          LABEL          MNEMONIC          OPERAND
000100        7A0700FF                      MOV.L            #00FFF000:32,ER7
              F000
000106        F800                          MOV.B            #00:8,R0L
000108        F900                          MOV.B            #00:8,R1L
00010A        8802                          ADD.B            #02:8,R0L
00010C        8901                          ADD.B            #01:8,R1L
00010E        A90A                          CMP.B            #0A:8,R1L
000110        46F8                          BNE              00010A:8
000112        6A881000                      MOV.B            R0L,@1000:16
000116        40EE                          BRA              000106:8
```

4.2.4 Searching Trace Information

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

Operation	**Console Message**
Enter TRACE_SEARCH A=116 (RET), and the console will only display those parts of the trace information in which the address bus value is H'116.	:TRACE_SEARCH A=116 (RET)

```
      BP         AB       DB    MA   R/W  ST    IRQ       NMI  RA   PROG       CLK
  -D'04077   00000116   40EE  EXT   R   PRG   11101111   1   11   11111111   06
  -D'03964   00000116   40EE  EXT   R   PRG   11101111   1   11   11111111   06
  -D'03851   00000116   40EE  EXT   R   PRG   11101111   1   11   11111111   06
                 :                             :                       :
```

4.2.5 Sequential Software Break

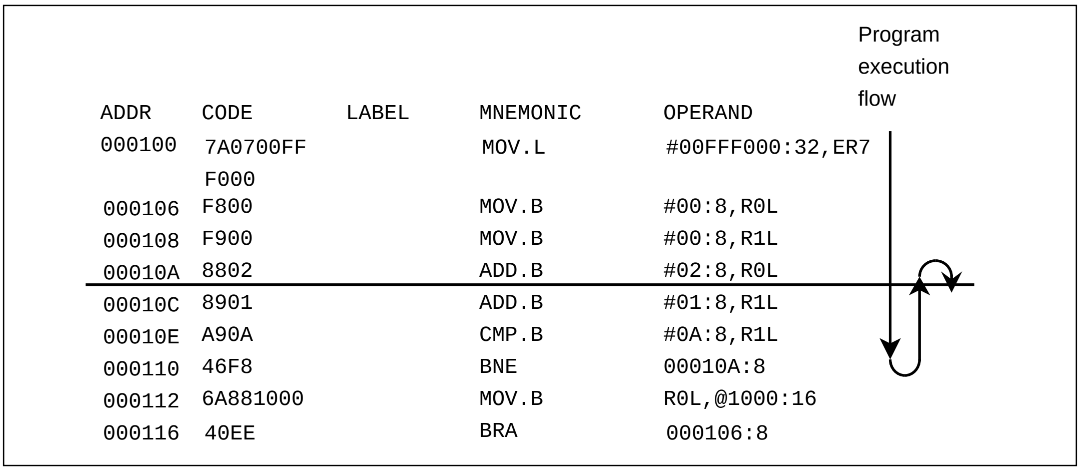
A break can be generated when specified addresses are passed in a specified order, using the BREAK_SEQUENCE command as follows:

Operations

Console Message

1. Enter BREAK_SEQUENCE 110 10A (RET), which terminates program execution when the instructions at addresses H'110 and H'10A are executed consecutively in that order, as shown in figure 4-1.

```
:BREAK_SEQUENCE 110 10A (RET)
```



```
ADDR    CODE        LABEL     MNEMONIC      OPERAND
000100  7A0700FF              MOV.L         #00FFF000:32,ER7
        F000
000106  F800                  MOV.B         #00:8,R0L
000108  F900                  MOV.B         #00:8,R1L
00010A  8802                  ADD.B         #02:8,R0L
00010C  8901                  ADD.B         #01:8,R1L
00010E  A90A                  CMP.B         #0A:8,R1L
000110  46F8                  BNE           00010A:8
000112  6A881000              MOV.B         R0L,@1000:16
000116  40EE                  BRA           000106:8
```

Figure 4-1 Program Execution Flow

2. Enter GO 100 (RET) to execute the program. When the specified condition is satisfied, execution terminates, and the console displays the data shown on the right. +++:BREAK SEQUENCE shows that execution has terminated because the condition specified in the BREAK_SEQUENCE command has been satisfied.

```
:GO 100 (RET)
** PC = xxxxxxxx
PC =0000010A   CCR =A9:I*H*N**C  EXR=07:*****210
MACH=00000000   MACL=00000000
ER0 - ER3   00000002  00000001  00000000  00000000
ER4 - ER7   00000000  00000000  00000000  00FFF000
RUN - TIME = D'0000H:00M:00S:000018US
+++:BREAK SEQUENCE
:_
```

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

```
:TRACE  (RET)
```

```
     IP        ADDR       LABEL   MNEMONIC     OPERAND
*-D'00009    00000100             MOV.L        #00FFF000:32,ER7
*-D'00008    00000106             MOV.B        #00:8,R0L
*-D'00007    00000108             MOV.B        #00:8,R1L
*-D'00006    0000010A             *  BREAK  *
*-D'00005    0000010A             ADD.B        #02:8,R0L
*-D'00004    0000010C             ADD.B        #01:8,R1L
*-D'00003    0000010E             CMP.B        #0A:8,R1L
*-D'00002    00000110             *  BREAK  *
*-D'00001    00000110             BNE          00010A:8
* D'00000    0000010A             *  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 the H8S series microcomputers, H8S/2655 series and H8S/2245 series (collectively referred to as MCU).

In addition to a high-speed CPU, the H8S/2655 series contains internal RAM, internal ROM, timers, watchdog timers, a serial communication interface, DMAC, DTC, I/O ports, and A/D and D/A converters on the same chip.

In addition to a high-speed CPU, the H8S/2645 series contains internal RAM, internal ROM, timers, watchdog timers, a serial communication interface, DTC, I/O ports, and A/D and D/A converters on the same chip.

Table 1-1 shows the functions of the MCU.

Table 1-1 MCU Functions

	Supported Device		
	H8S/2655 Series		**H8S/2245 Series**
Function	**H8S/2653**	**H8S/2655**	**H8S/2245**
Maximum address memory size that can be managed	16 Mbytes	16 Mbytes	16 Mbytes
Internal ROM	64 kbytes	128 kbytes	128 kbytes
Internal RAM	4 kbytes	4 kbytes	4 kbytes
DMAC	Built-in	Built-in	—
DTC	Built-in	Built-in	Built-in
16-bit timer pulse unit	6 channels	6 channels	3 channels
Programmable pulse generator	Built-in	Built-in	—
8-bit timer	2 channels	2 channels	2 channels
Watchdog timer	Built-in	Built-in	Built-in
Serial communication interface	3 channels	3 channels	3 channels
A/D converter	10 bits x 8 channels	10 bits x 8 channels	10 bits x 4 channels
D/A converter	8 bits x 2 channels	8 bits x 2 channels	8 bits x 2 channels
Interrupt controller External interrupts	9	9	9
Internal interrupt sources	52	52	34
Package (plastic)	TFP-120 or FP-128	TFP-120 or FP-128	FP-100B or TFP-100B

The emulator operates in just the same way as the MCU 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 an emulator pod, and a user system interface cable. The emulator pod should be connected directly to the user system.

In this manual, the MCU is used to refer collectively to the H8S/2655 series and H8S/2245 series when there is no difference in their characteristics.

1.2 Specification

The main features of the emulator are its emulation functions, its floppy disk utility functions, and its host computer interface functions, as listed in tables 1-2 to 1-4, respectively. To be specific, table 1-4 lists the functions for data transfer between the E7000PC and its host computer, the IBM PC.

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 20 MHz at max. • Executes until a hardware or software 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) • Measures the execution time between the specified points. • Parallel mode: Displays trace data and modifies memory contents during emulation.	7.2.21
	EXECUTION_ MODE	Specifies execution mode.	7.2.19

Table 1-2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Break condition setting	BREAK_ CONDITION1,2,3, 4,5,6	Sets hardware break conditions (1). • Normal break: Breaks when the following condition is satisfied (up to two points): — Address bus or data bus value — Read/write condition — External probe values — External interrupts — NOT condition — Delay count — Satisfaction count • Specification of the satisfaction sequence up to four points Sets hardware break conditions (2). • Program counter (PC) value • Address bus value • Read/write condition	7.2.7
	BREAK	Sets software break conditions. • Normal break: Sets up to 255 breakpoints. • Sets pass count.	7.2.6
	BREAK_ SEQUENCE	• Sequential software break up to four points and one reset point	7.2.8

Table 1-2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Trace data acquisition and display	TRACE	• Displays executed instruction mnemonic. • Displays the following data for each bus cycle: — Instruction mnemonic — Address bus and data bus values — Access area and status — MCU I/O control signals — External probe value (eight probes) — Clock count	7.2.46
	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 bus and data bus values — Read/write condition — Access type — External probe value (eight probes) — System control signals — NOT condition — Delay count • Outputs a low pulse from the trigger output pin when a condition is satisfied. — Address bus and data bus value — Read/write condition — Access type — External probe value (eight probes) — System control signals — NOT condition — Delay count	7.2.47
	TRACE_SEARCH	Searches for trace data.	7.2.52
	TRACE_MEMORY	Specifies memory address to trace.	7.2.49
	TRACE_MODE	Specifies refresh cycle display.	7.2.50

Table 1-2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Single-step execution	STEP	• Executes one step at a time, and displays the following. — Instruction mnemonic — Memory contents — Register contents • Executes displaying the above data after a branch instruction 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.	7.2.42
	STEP_ INFORMATION	Specifies information to be displayed during single-step execution.	7.2.43
	STEP_OVER	Executes subroutine as a single step.	7.2.44
Memory access	MEMORY, DUMP	Displays or modifies memory contents in 1-, 2-, or 4-byte units.	7.2.28, 7.2.17
	MAP	Specifies memory attributes in a minimum of 128-kbyte units. • User memory • Write-protected area (a minimum of 128-kbyte units) • Guarded memory area (a minimum of 512-kbyte units) • Emulation memory 512-kbyte standard (SRAM with no wait state) 1-Mbyte or 4-Mbyte option (SRAM with 3 states + 1 wait state. Memory attributes can be specified in 1-Mbyte or 2-Mbyte units.)	7.2.27
	FILL	Writes data in specified pattern.	7.2.20
	DATA_SEARCH, DATA_CHANGE	Searches for and replaces data in specified pattern.	7.2.14, 7.2.13
Clock selection	CLOCK	• Selects emulator internal clock. 8 MHz or 20 MHz • Selects user system clock. 2 MHz to 30 MHz • Selects crystal oscillator in emulator pod. 10 MHz to 20 MHz	7.2.10

Table 1-2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Register access	REGISTER, .<register name>	Displays and modifies MCU register contents.	7.2.36, 7.2.1
Line assembly	ASSEMBLE	Assembles instruction mnemonics and modifies memory contents. • Enables use of labels and symbol names.	7.2.5
Disassembly	DISASSEMBLE	Disassembles memory contents. • Displays labels and symbol names.	7.2.15
Execution time, pass count measurement	GO	Measures GO command execution time. • Measures total run-time (approx. 305 hours max). • Measures execution time during a specified range.	7.2.21
	PERFORMANCE_ ANALYSIS	Measures execution time and pass count of the specified range.	7.2.32
	TRACE	Counts clock in each bus cycle.	7.2.46
Test functions	FILL	Reads or writes the specified data to the memory.	7.2.20
	CHECK	Tests MCU I/O signals.	7.2.9
	DISPLAY_ COVERAGE, SET_COVERAGE	Traces C0 coverage. Traces the addresses executed by the MCU during user program execution.	7.2.16, 7.2.39
Symbolic debugging	LOAD, INTFC_LOAD	Loads symbols from host computer.	9.4.2, 9.4.7
	SYMBOL, SHORT_SYMBOL	Defines, cancels, and displays symbols.	7.2.45, 7.2.40
	!<symbol name> &<symbol name>	Displays symbol contents according to attributes associated with symbol names. • Function names and 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.	7.2.11
	RADIX	Enables value input in binary, octal, decimal, hexadecimal, or ASCII characters. (Default can be specified.)	7.2.35

Table 1-2 Emulation Functions (cont)

Command Type	Command	Function	Reference Section
Results display	PRINT	Outputs to printer or file.	7.2.33
	RESULT	Displays emulation results.	7.2.38
Others	MOVE, MOVE_TO_RAM	Transfers memory contents. • Memory to memory • ROM (user system memory) to emulation memory	7.2.30, 7.2.31
	CONVERT	Converts number display. Displays in binary, octal, decimal, hexadecimal, or ASCII characters.	7.2.12
	STATUS	Displays emulator operating status.	7.2.41
	GO	Monitors emulation. Monitors emulation status every 200 ms and displays abnormalities found during emulation.	7.2.21
	RESET	Inputs RES signal to MCU.	7.2.37
	HELP	Displays all commands.	7.2.22
	HISTORY	Displays the history of the input command.	7.2.23

Table 1-3 Floppy Disk Utility Functions

Command Type	Command	Function	Reference Section
Backup	B (Monitor command)	Backs up and verifies floppy disk.	3.6 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
Free area display	FLOPPY_CHECK	Displays free area on floppy disk.	8.4.10

Table 1-4 Host Computer Interface Functions

Command Type	Command	Function	Reference Section
Interface condition setting	HOST	Sets the following for RS-232C interface: • Transfer rate (2400 to 38400 bps) • Data length (seven or eight bits) • Parity (even, odd, or none) • Number of stop bits (one or two 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, LOAD	Loads memory contents from host computer. (only for E7000)	9.4.2, 9.4.7, 11.3.1
	VERIFY, INTFC_VERIFY, VERIFY	Verifies memory contents against host computer file. (only for E7000)	9.4.6. 9.4.10, 11.3.3
	SAVE, INTFC_SAVE, SAVE	Saves memory contents in host computer. (only for E7000)	9.4.3, 9.4.8, 11.3.2
Data transfer to and from floppy disk	TRANSFER, INTFC_TRANSFER	• Receives data from host computer and writes it to file. • Transfers file contents to host computer.	9.4.5, 9.4.9
Host computer terminal selection	TERMINAL	Lets the host computer operate as a terminal.	9.4.4

1.3 Realtime Emulation

The emulator enables realtime emulation 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 MCU at a specified period
- Parallel mode: Enables the user to display and modify memory contents and display trace information during user program execution

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

1.3.1 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 from the trigger output probe. When a number of times or sequential break for the software break condition is specified, 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.

1.3.2 Cycle Reset Mode

Cycle Reset Mode Function: The emulator inputs the RES signal to the MCU 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 MCU. This function is useful for observing waveforms from an initial state such as power-on reset to the specified time.

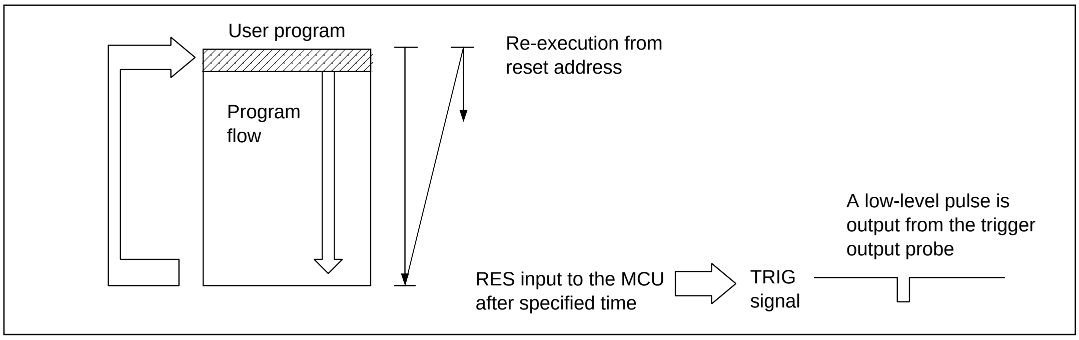

User program
Program flow
Re-execution from reset address
A low-level pulse is output from the trigger output probe
RES input to the MCU after specified time
TRIG signal

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 immediately before the RES signal input 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 MCU when the time specified by a command passes, regardless of the MCU operating mode.

Figure 1-2 shows the timing for output from the trigger output probe in cycle reset mode.

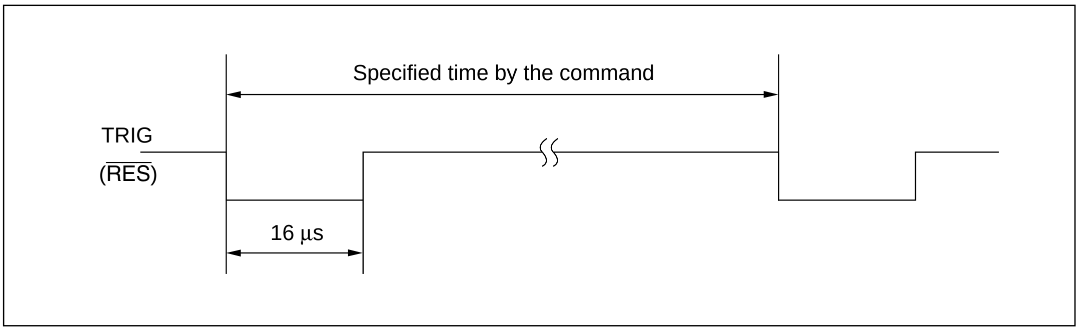


Figure 1-2 Trigger Signal Output Timing

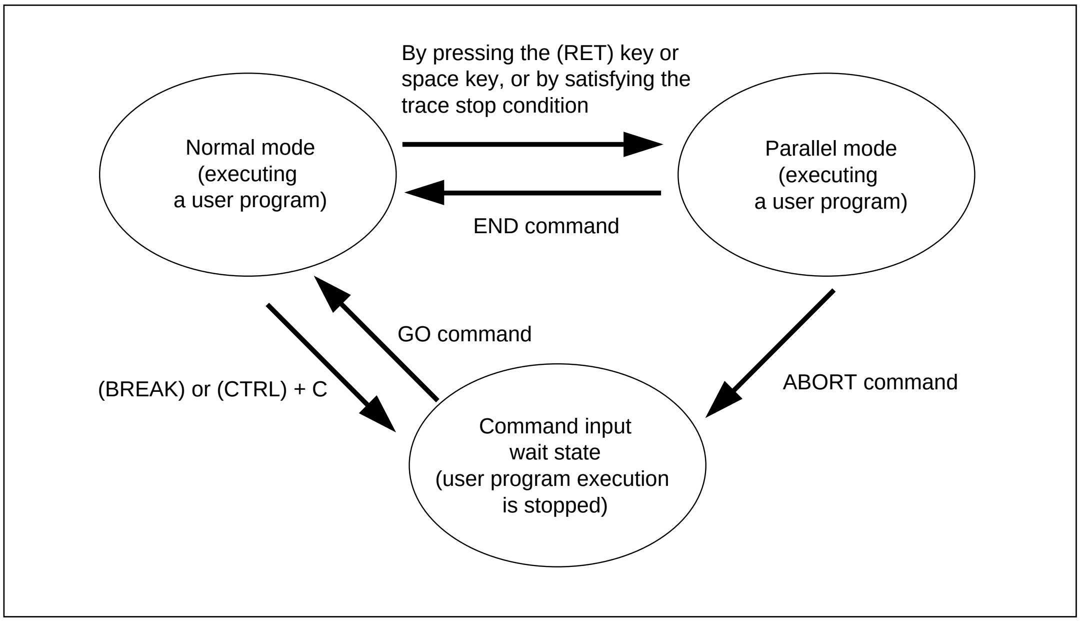
1.3.3 Parallel Mode

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

Parallel Mode Specification: Parallel mode can be activated during GO command realtime emulation by any of the following methods as shown in figure 1-3.

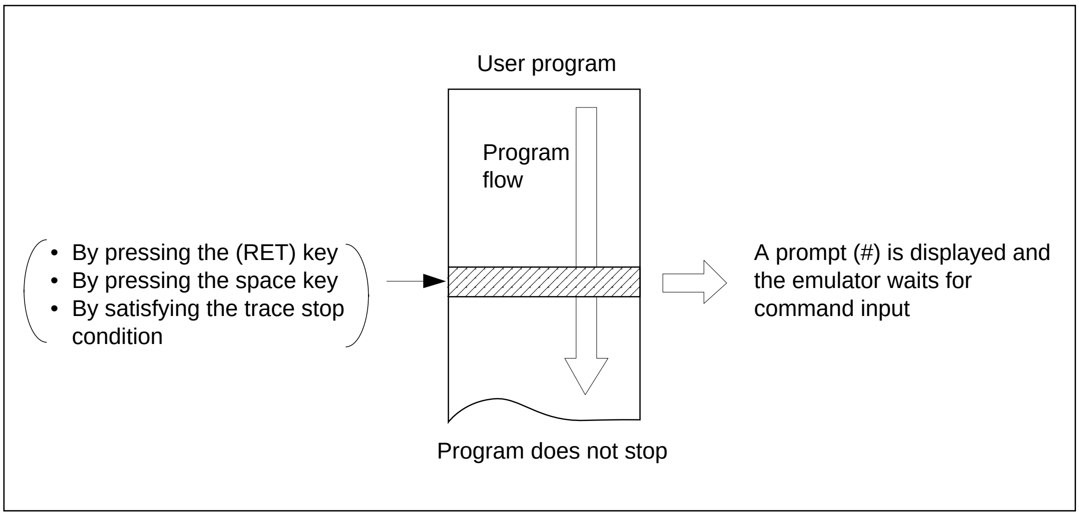
- Press the (RET) key
- Press the space key
- Satisfy a trace stop condition specified by the TRACE_CONDITION command

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 1-4).
Input the END (E) command to return to the normal mode. Input the ABORT (AB) command to stop user program execution in the parallel mode.



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

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. Parallel mode continues even after the command execution is terminated.
  - The END command terminates 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.

Commands usable in parallel mode are listed in table 7-1.

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

- **Standard emulation memory or internal ROM/RAM**
  When accessing standard emulation memory or internal ROM/RAM in parallel mode, the user program temporarily halts. This pause lasts for six states (one byte access) when accessed by the MEMORY command. Therefore, realtime emulation cannot be performed.
- **Optional emulation memory**
  When accessing optional emulation memory, the user program does not stop. However, an access to the area allocated to optional emulation memory is executed in 3 states + 1 wait state.
- **Internal I/O or user memory**
  When accessing internal I/O or user memory, the user program temporarily halts. This pause lasts for about 3.0 ms (one byte access) at 20-MHz clock operation. Therefore, realtime emulation cannot be performed.
- **In the above three 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:

- **Cycle reset mode (R option of GO command)**
- **Time measurement mode1 or 2 (I1 or I2 option of GO command)**

1.4 Break Function

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

• Hardware break:	Caused by the MCU's signal status as specified
• Software break:	Caused by the program counter (PC)
• 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

A hardware break can be specified using the BREAK_CONDITION1 to 6 commands. Specifiable break conditions are listed in table 1-5.

Table 1-5 Specifiable Hardware Break Conditions

Specification	BREAK_ CONDITION 1	BREAK_ CONDITION 2,3,4	BREAK_ CONDITION 5,6
Address bus value	O	O	O
Data bus value	O	O	—
Read/write	O	O	O
Access type	O	O	—
External probe value	O	O	—
External interrupts	O	O	—
NOT condition	O	—	—
Satisfaction count	O	—	—
Delay count	O	—	—
PC value	—	—	O

O: Specifiable

Note: The MCU prefetches instructions. When a hardware break condition is satisfied in a prefetch cycle, a hardware break may occur before the prefetched instructions are executed.

Address Bus Value: Address bus values can be specified with the BREAK_CONDITION1,2,3,4 commands. A break occurs when the MCU address bus value matches the specified condition. A particular address or an address range can be specified.

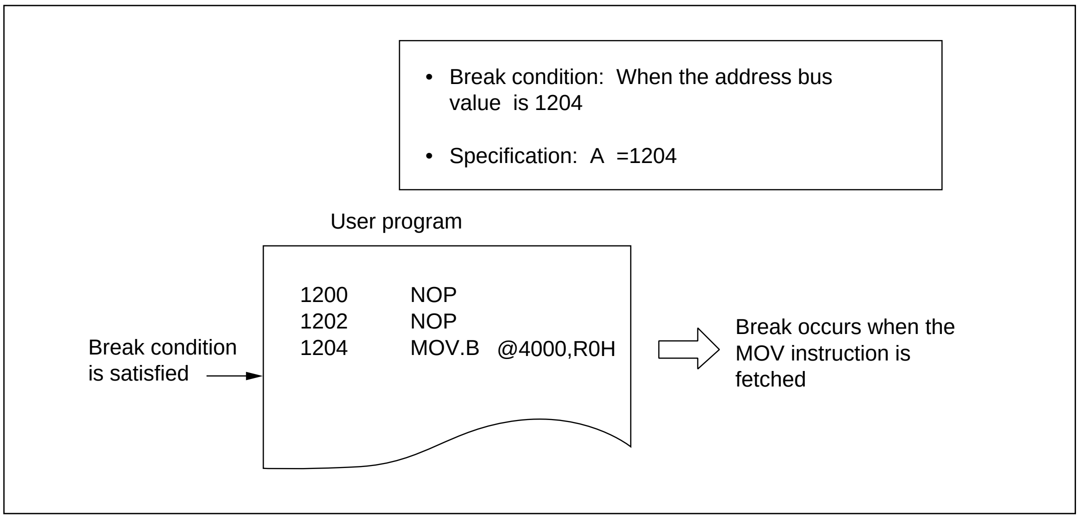

• Break condition: When the address bus value is 1204
• Specification: A =1204
User program
1200 NOP
1202 NOP
1204 MOV.B @4000,R0H
Break condition is satisfied
Break occurs when the MOV instruction is fetched

Figure 1-5 Break with Address Bus Value

Data Bus Value: Data bus values can be specified with the BREAK_CONDITION1,2,3,4 commands. A break occurs when the MCU data bus value matches the specified condition. The emulator checks both program fetch and data access for the condition.

The data size must be selected from word access (WD) or byte access (HD, LD, or D).

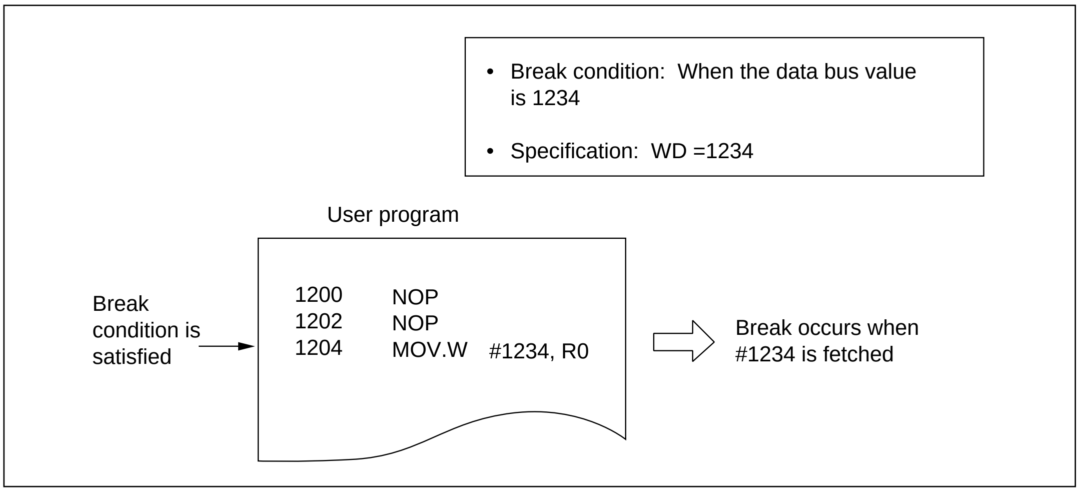
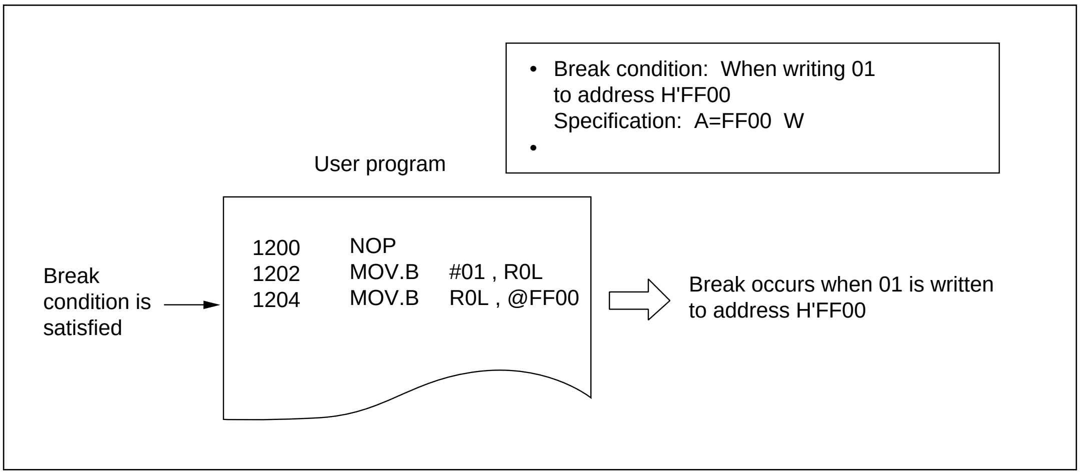

• Break condition: When the data bus value is 1234
• Specification: WD =1234
User program
1200 NOP
1202 NOP
1204 MOV.W #1234, R0
Break condition is satisfied
Break occurs when #1234 is fetched

Figure 1-6 Break with Data Bus Value

Read/Write Condition: Read/write condition can be specified with the BREAK_CONDITION1,2,3,4 commands. A break occurs when the MCU’s RD and WR (HWR and LWR) 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'FF00
Specification: A=FF00 W
•
User program
1200 NOP
1202 MOV.B #01 , R0L
1204 MOV.B R0L , @FF00
Break condition is satisfied
Break occurs when 01 is written to address H'FF00

Figure 1-7 Break with Read/Write Condition

External Probe Value: External probe value can be specified with the BREAK_CONDITION1,2,3,4 commands.

- A break occurs when external probe signal levels match the specified value. For example, the user can specify levels of probes 1 and 2 to high and the other probe's levels to low to generate a break.

- Multibreak function
  External probe no.8 has a falling-edge detection function. By using this function, you can generate a break in other emulators of the same type simultaneously. Whether to use external probe no.8 as a level signal input pin or a falling-edge detection pin can be selected by the EXECUTION_MODE command. The following explains how to perform a multibreak with the external probe.

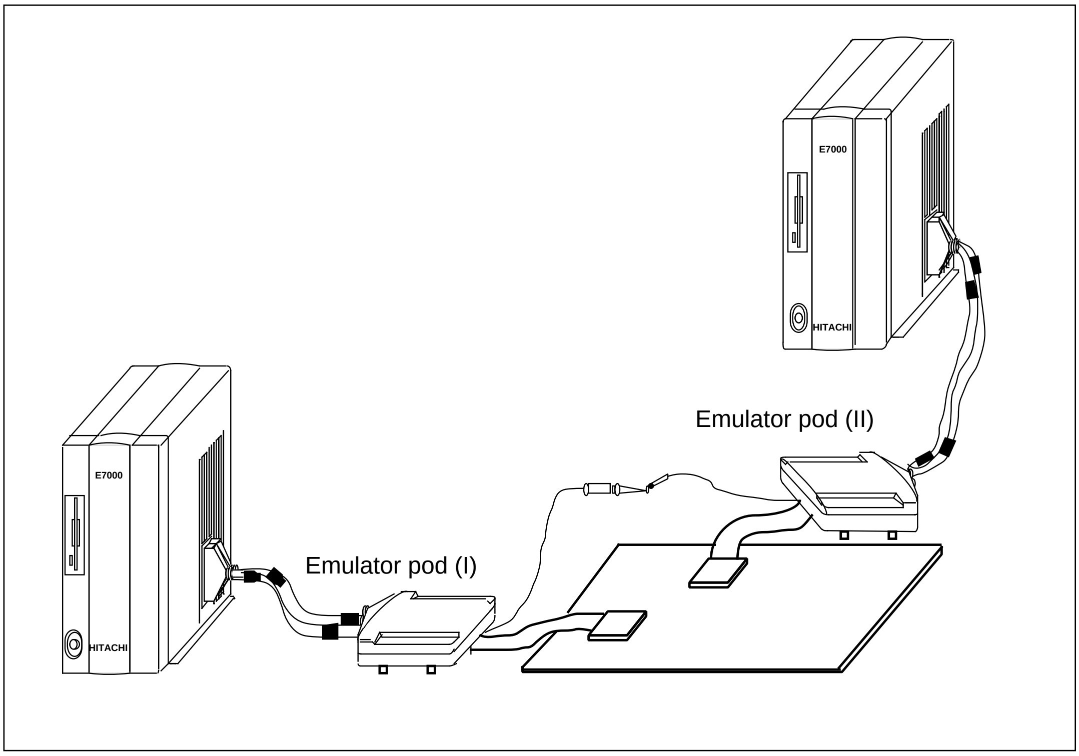


Figure 1-8 Multibreak Function

Procedure

1. Connect external probe no.8 of emulator pod (I) to the trigger output probe of emulator pod (II).

2. Set the break condition for emulator pod (II).

3. Set the break condition for external probe no.8 of emulator pod (I). At this time, external probe no.8 must be specified to break at a low level. (PROB=B'0xxxxxxx)

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 a pulse output from the trigger output probe. However, the break in emulator pod (I) follows that in emulator pod (II) by several bus cycles.

External Interrupt Condition: External interrupt (NMI and IRQ0 to IRQ7) condition can be specified with the BREAK_CONDITION1,2,3,4 commands. Emulation stops when the value of the interrupt signal matches the specified value. Figure 1-9 shows an example.

- NMI: Low or high level can be specified.
- IRQ0 to IRQ7: Low or high level can be specified.

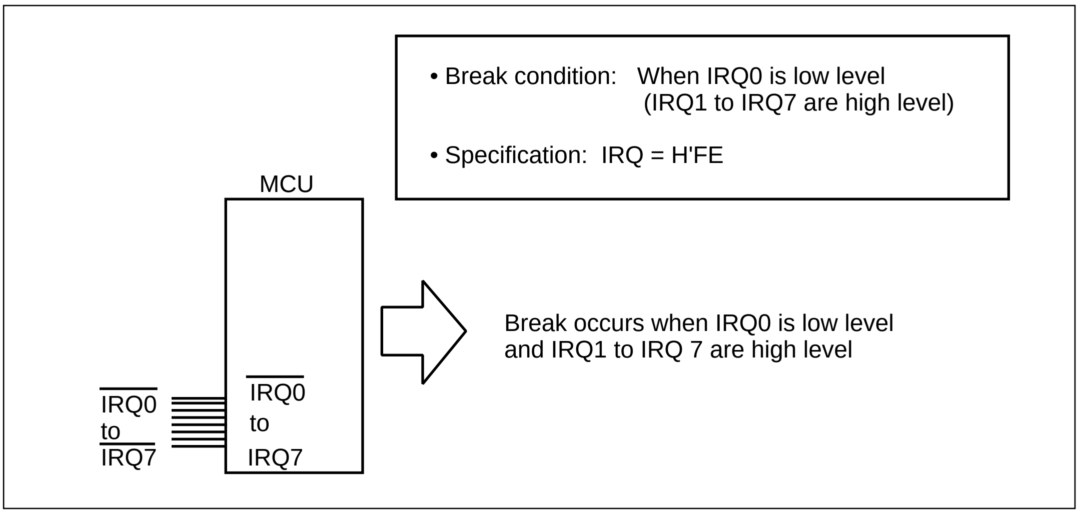

• Break condition: When IRQ0 is low level
(IRQ1 to IRQ7 are high level)
• Specification: IRQ = H'FE
MCU
Break occurs when IRQ0 is low level
and IRQ1 to IRQ 7 are high level
IRQ0
to
IRQ7
IRQ0
to
IRQ7

Figure 1-9   Break with External Interrupt

Notes: 1. External interrupt signals (IRQ0 to IRQ7) can also be used for other input depending on the MCU control register settings. However, the emulator monitors signal status where the external interrupt signals are allocated regardless of the control register settings.

2. A break using the external interrupt (IRQ0 to IRQ7) condition can be used only in the H8S/2655 series; it cannot be used in the H8S/2245 series.

NOT Condition (BREAK_CONDITION1): A break occurs when the break condition of the address bus and data bus value is not satisfied. The user can specify this condition only with the BREAK_CONDITION1 command.

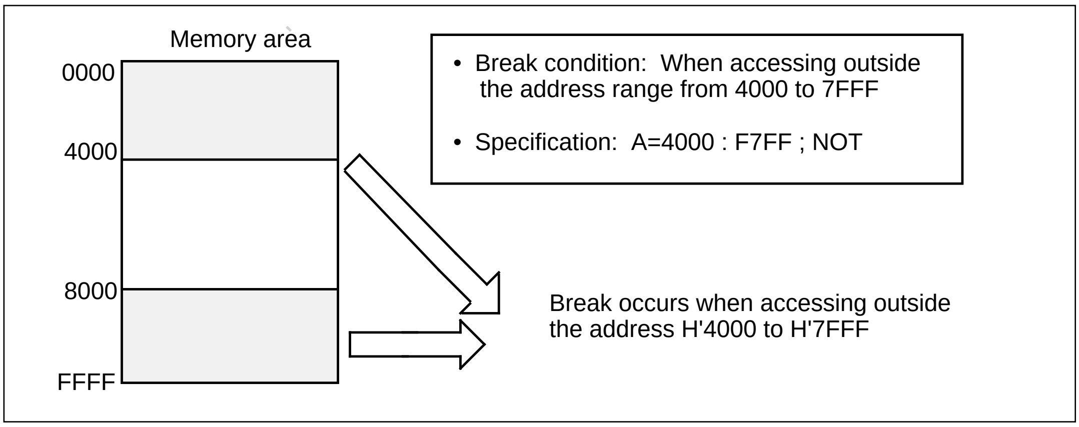

Memory area
0000
4000
8000
FFFF
• Break condition: When accessing outside the address range from 4000 to 7FFF
• Specification: A=4000 : F7FF ; NOT
Break occurs when accessing outside the address H'4000 to H'7FFF

Figure 1-10 Break with NOT Condition

Number of Times Break Condition is Satisfied: This function can only be specified with the BREAK_CONDITION1 command. A break occurs when the above break condition is satisfied for a specified number of times (4,096 max). When specifying this condition, specify it in combination with any of the above break conditions.

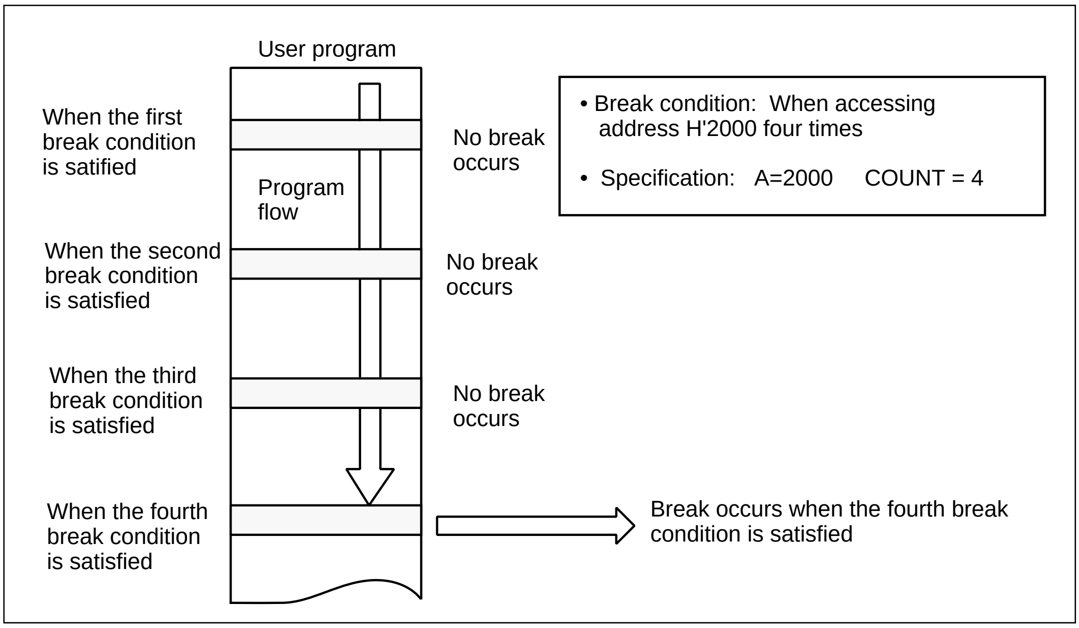

User program
When the first break condition is satified
Program flow
No break occurs
When the second break condition is satisfied
No break occurs
When the third break condition is satisfied
No break occurs
When the fourth break condition is satisfied
Break occurs when the fourth break condition is satisfied
• Break condition: When accessing address H'2000 four times
• Specification: A=2000 COUNT = 4

Figure 1-11 Break with Number of Times Break Condition is Satisfied Specification

Delay Count: This function can only be specified with the BREAK_CONDITION1 command. A break occurs when the above break condition is satisfied and the emulator executes the bus cycle for a specified number of times (32,767 max). When specifying this condition, specify it in combination with any of the above break conditions.

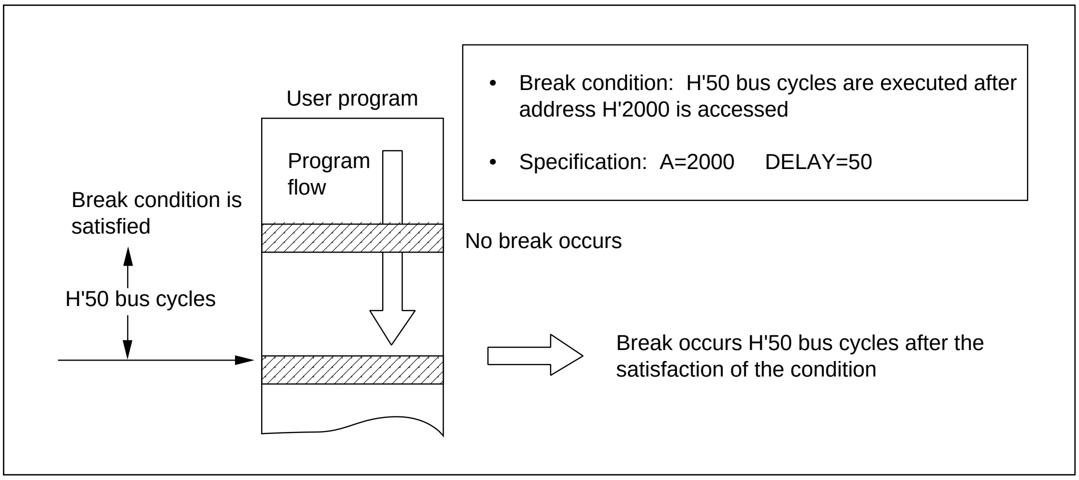

User program
Program flow
Break condition is satisfied
H'50 bus cycles
• Break condition: H'50 bus cycles are executed after address H'2000 is accessed
• Specification: A=2000 DELAY=50
No break occurs
Break occurs H'50 bus cycles after the satisfaction of the condition

Figure 1-12 Break with Delay Count Specification

PC Value: PC value condition can be specified with the BREAK_CONDITION5,6 commands. A break occurs when the MCU program counter (PC) value satisfies the specified condition.

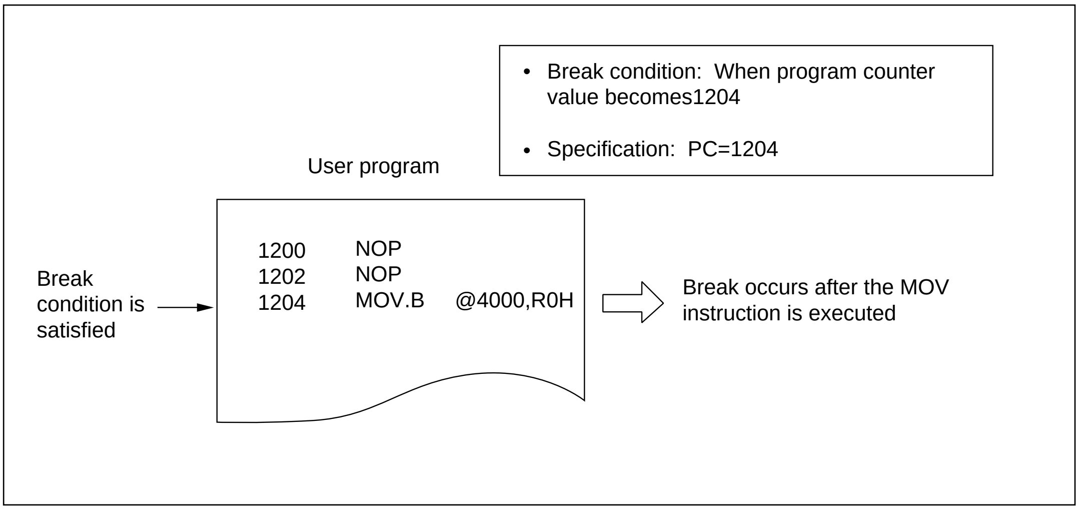

• Break condition: When program counter value becomes1204
• Specification: PC=1204
User program
1200 NOP
1202 NOP
1204 MOV.B @4000,R0H
Break condition is satisfied
Break occurs after the MOV instruction is executed

Figure 1-13 Break with PC Value Specification

Sequential Break Condition: Sequential break condition can be specified with the BREAK_CONDITION1,2,3,4 commands. In sequential break mode, a break occurs when hardware break conditions 4 to1 have been satisfied in that order. There are the following three types of sequential break.

- Sequential break mode 1

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

- Sequential break mode 2

  When break conditions 3, 2, and 1 are satisfied in that order, a break occurs.

- Sequential break mode 3

  When break conditions 4, 3, 2, 1 are satisfied in that order, a break occurs.

  Specify the break condition with the BREAK_CONDITION1,2,3,4 commands. The user can specify any of the above conditions.

  To use the sequential break function, specify one of sequential options (;S1 to ;S3) with the mode option of the GO command. If this option is not specified, a break occurs when each break condition is satisfied and sequential break does not occur.

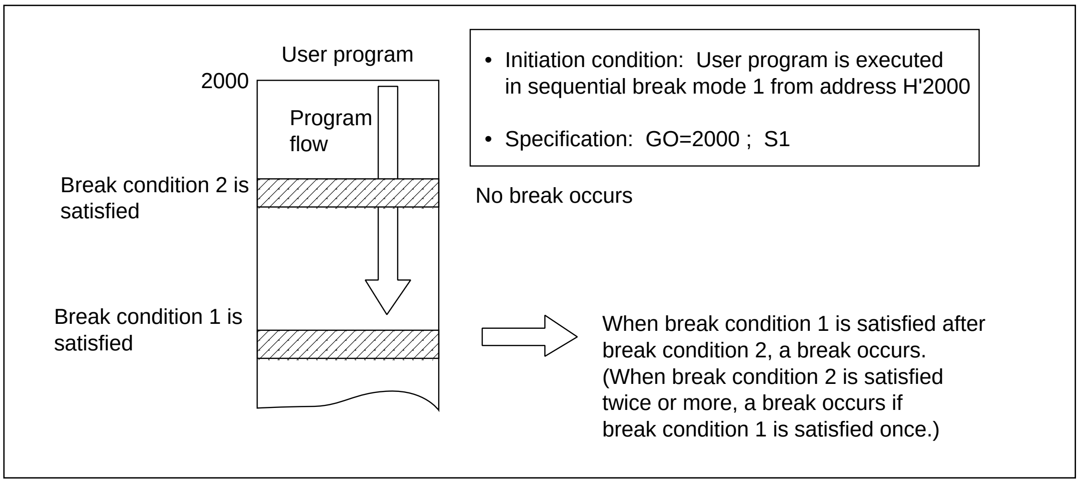

User program
2000
Program flow
Break condition 2 is satisfied
Break condition 1 is satisfied
• Initiation condition: User program is executed in sequential break mode 1 from address H'2000
• Specification: GO=2000 ; S1
No break occurs
When break condition 1 is satisfied after break condition 2, a break occurs. (When break condition 2 is satisfied twice or more, a break occurs if break condition 1 is satisfied once.)

Figure 1-14 Break with Sequential Specification

Break Timing: Hardware break sampling timing synchronizes with the MCU bus cycle (AS signal).

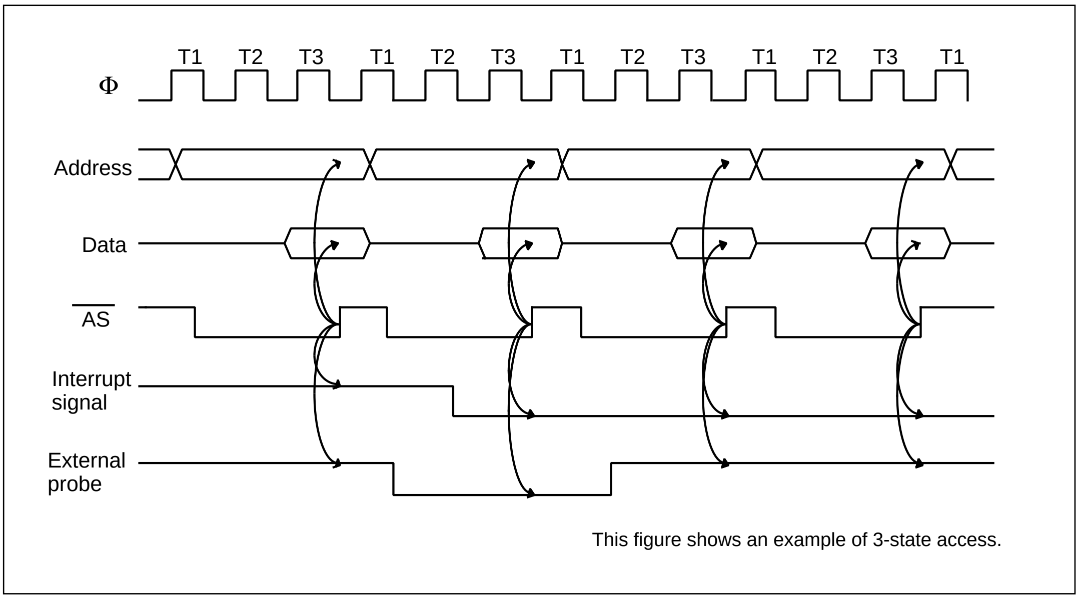


Figure 1-15 Break Timing

Notes: 1. Because the MCU prefetches instructions, a break may occur before the prefetched instruction is executed when a break condition is satisfied during a prefetch cycle.

2. When the rising edge of the AS signal (sampling timing) overlaps 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.

1.4.2 Software Break

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

After the GO command is executed, the contents at the specified address will be replaced with a break instruction and the user program will be executed. When user program execution stops, the break instruction will be replaced again with the contents at the specified address. Therefore, the contents at the specified address can be accessed immediately after user program execution stops, using the DISASSEMBLE or DUMP command. However, since the replacement is performed after program execution stops, when accessing the memory contents in parallel mode, the contents at the specified address will still be replace with a break instruction.

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

- Normal break
- Sequential break

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 4095 (H'FFF).

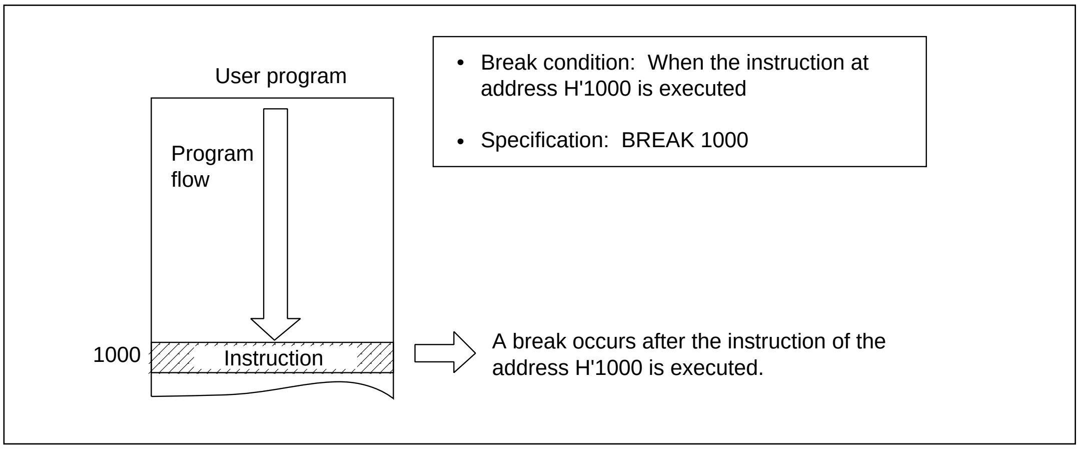
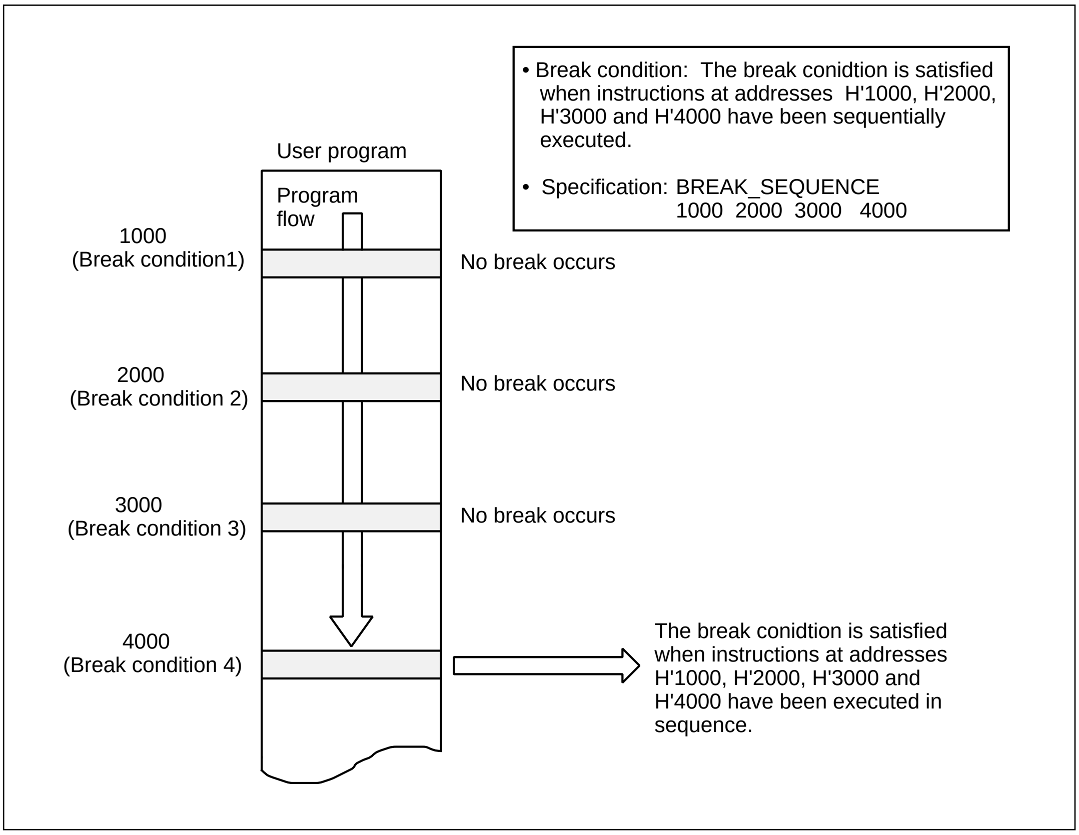

User program
Program flow
1000
Instruction
• Break condition: When the instruction at address H'1000 is executed
• Specification: BREAK 1000
A break occurs after the instruction of the address H'1000 is executed.

Figure 1-16 Normal Break (Software Break)

Note: When specifying the number of times the break condition is 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.

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

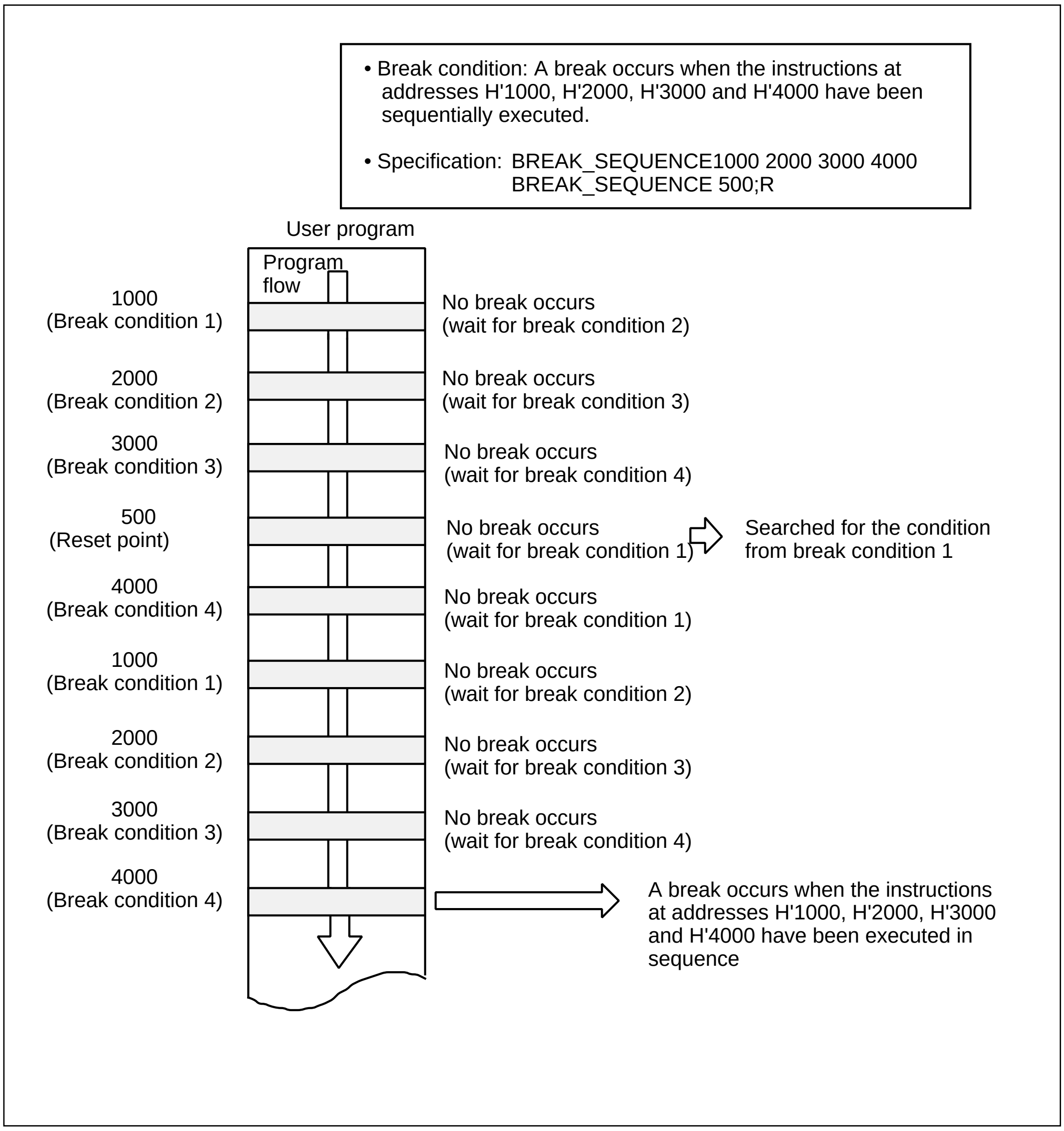
Figure 1-17 illustrates the normal sequential break and figure 1-18 describes a sequential break when a reset point is specified.



• Break condition: The break conidtion is satisfied when instructions at addresses H'1000, H'2000, H'3000 and H'4000 have been sequentially executed.
• Specification: BREAK_SEQUENCE 1000 2000 3000 4000
User program
Program flow
1000 (Break condition1)
No break occurs
2000 (Break condition 2)
No break occurs
3000 (Break condition 3)
No break occurs
4000 (Break condition 4)
The break conidtion is satisfied when instructions at addresses H'1000, H'2000, H'3000 and H'4000 have been executed in sequence.

Figure 1-17 Sequential Break

Note: When specifying the sequential break, emulator firmware performs processing every time the program passes a break condition address. As a result, the program will not operate in realtime.



• Break condition: A break occurs when the instructions at addresses H'1000, H'2000, H'3000 and H'4000 have been sequentially executed.
• Specification: BREAK_SEQUENCE1000 2000 3000 4000
BREAK_SEQUENCE 500;R
User program
Program flow
1000 (Break condition 1)
No break occurs (wait for break condition 2)
2000 (Break condition 2)
No break occurs (wait for break condition 3)
3000 (Break condition 3)
No break occurs (wait for break condition 4)
500 (Reset point)
No break occurs (wait for break condition 1)
Searched for the condition from break condition 1
4000 (Break condition 4)
No break occurs (wait for break condition 1)
1000 (Break condition 1)
No break occurs (wait for break condition 2)
2000 (Break condition 2)
No break occurs (wait for break condition 3)
3000 (Break condition 3)
No break occurs (wait for break condition 4)
4000 (Break condition 4)
A break occurs when the instructions at addresses H'1000, H'2000, H'3000 and H'4000 have been executed in sequence

Figure 1-18 Sequential Break (Reset Point Specification)

Note: When specifying the sequential break, emulator firmware performs processing every time the program passes a break condition address. As a result, the program will not operate in realtime.

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

1.5 Realtime Trace Function

The emulator can trace MCU external bus information during realtime emulation without affecting the user system. The emulator can fetch external bus information of the MCU address or data, and the external probe value up to 32,767 bus cycles. The TRACE command displays acquired trace information and allows the executed program flow to be checked.

Trace information:

- Address bus: 24 bits
- Data bus: 16 bits
- External probe: Eight
- MCU I/O control signals: 30
- Number of bus cycle clocks (ø): Seven bits (127 kbytes max)
- Memory contents tracing: 16 bits

The emulator displays trace information in the following methods:

- Displays only the instruction information in mnemonic.
- Displays the trace information in mnemonic in bus cycle units.
- Searches for the specified trace information and displays it with the TRACE_SEARCH command.

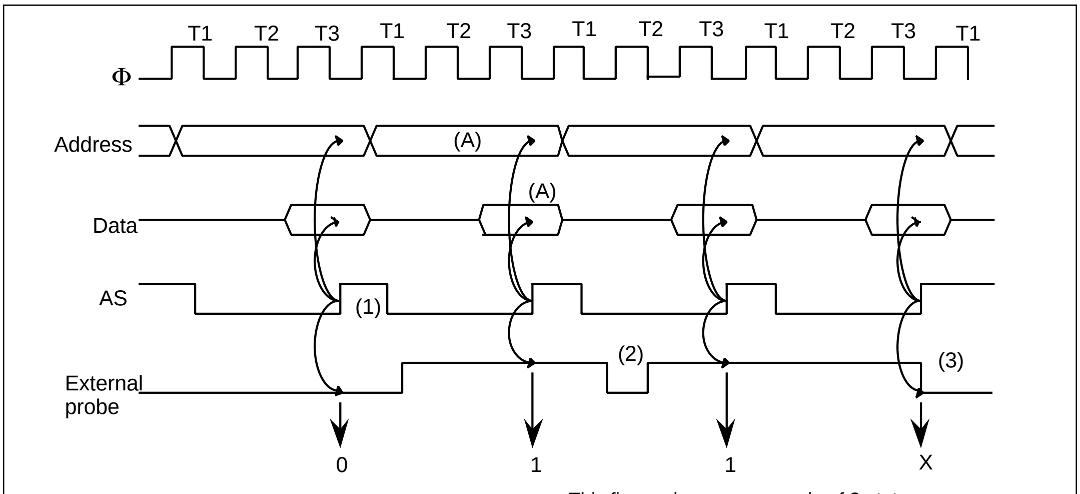
The emulator acquires bus information even in MCU single-chip mode.

1.5.1 Trace Timing

Trace information is acquired in trace memory synchronized with rising edges of the AS signal.

Because external probe signal input is not synchronized with the AS 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 (AS) cycles from the end of the previous bus cycle to the end of the current bus cycle. Figure 1-19 shows an example of the external probe signal trace.



This figure shows an example of 3-state access.

Figure 1-19 External Probe Signal Trace

Example:

- External probe signal
  - Trace information sampled at rising edges of AS (figure 1-19 (1)).
  - When the external probe signal changes between samplings, it cannot be reflected in the trace data (figure 1-19 (2)).
  - When a sampling edge coincides with a change in the external probe signal, the trace contents are unfixed (figure 1-19 (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 with the GO, STEP, or STEP_OVER command, trace is carried out continuously for a maximum of the latest 32,767 bus cycles until a break condition is satisfied. When no condition is specified with the TRACE_CONDITION command, the default is free trace. Figure 1-20 illustrates the free trace operation.

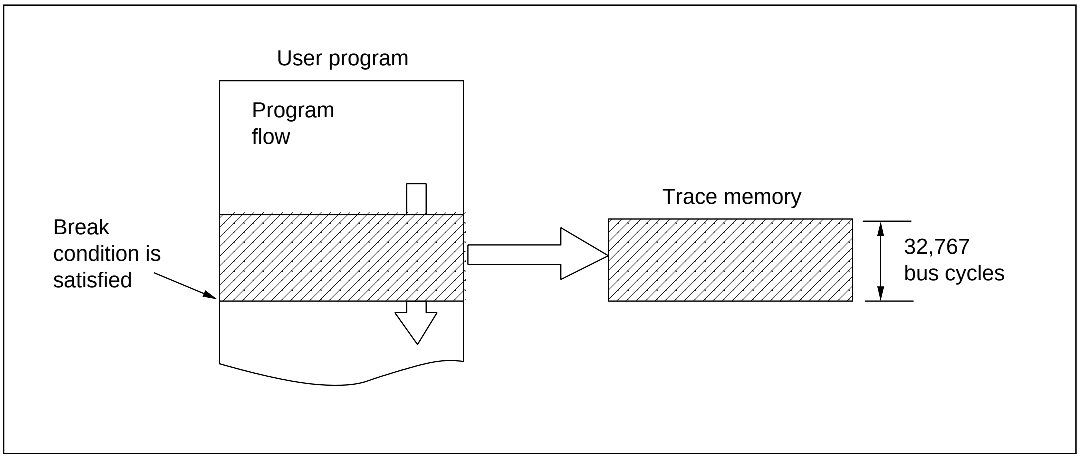


Figure 1-20 Free Trace Execution

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 1-21 illustrates the subroutine trace operation.

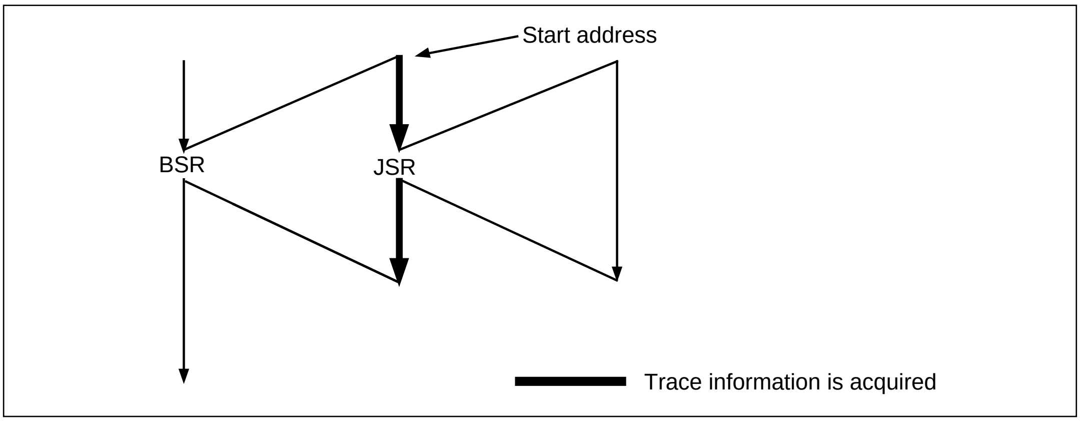


Figure 1-21 Subroutine Trace Specification

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

Figure 1-22 illustrates the range trace operation.

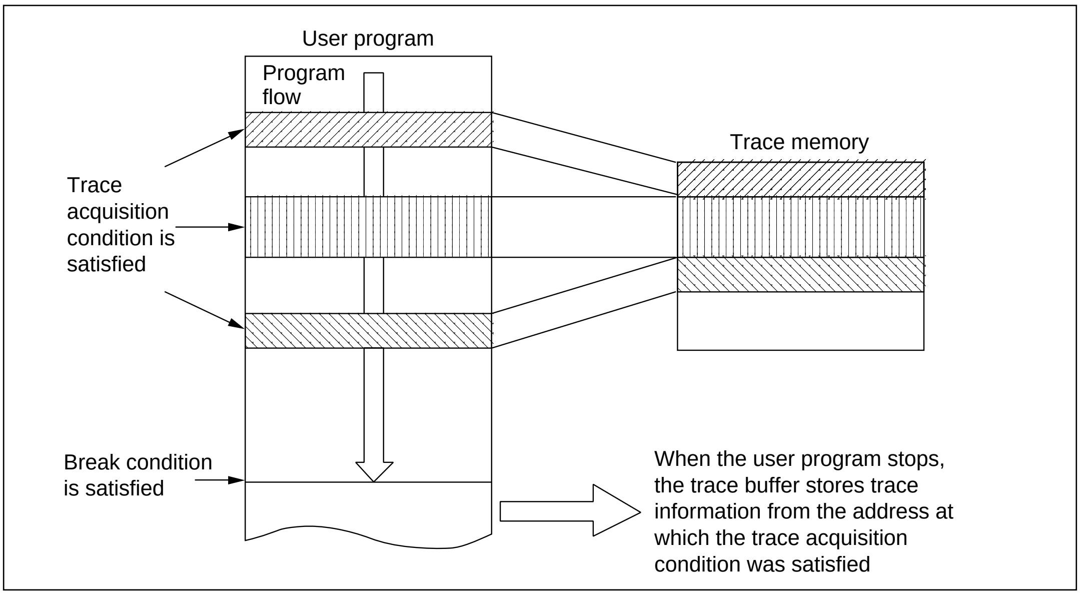

User program
Program flow
Trace memory
Trace acquisition condition is satisfied
Break condition is satisfied
When the user program stops, the trace buffer stores trace information from the address at which the trace acquisition condition was satisfied

Figure 1-22 Range Trace Specification

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 in parallel mode, although realtime emulation does not stop. Refer to section 1.3.3, Parallel Mode, for details.

Once the trace stop conditions have been satisfied and the trace information has been displayed, the user can specify the trace stop condition again. The user can specify the following conditions.

- Address bus and data bus values
- Read/write condition
- Access type (PRG, DAT, DMA, DTC)
- External probe value
- System control signal (BREQ, BACK)
- NOT condition
- Delay count (1 to H'7FFF)

Figure 1-23 illustrates the trace stop condition operation.

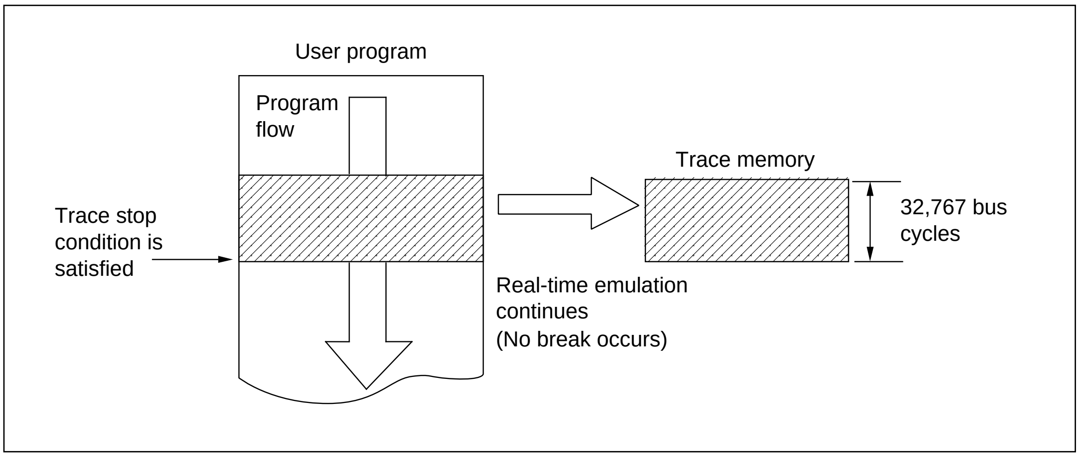


Figure 1-23 Trace Stop Condition Specification

1.5.3 Trace Display

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

Instruction Display: Only the executed instructions will be displayed in mnemonics.

Bus Cycle Display: Trace information is displayed in bus cycle units. When a trace is specified in special addresses, the memory contents are also displayed. The memory address is specified with the TRACE_MEMORY command.

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

1.6 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.

- MCU control registers (PC, CCR, EXR, MACH, MACL)
- MCU general-purpose registers (ER0 to ER7)
- Instruction address
- Instruction mnemonic
- Memory contents
- Termination cause

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 a single step, use the STEP_OVER command.

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

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

Bcc, BRA, BRN, JMP, BSR, JSR, RTS, RTE

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

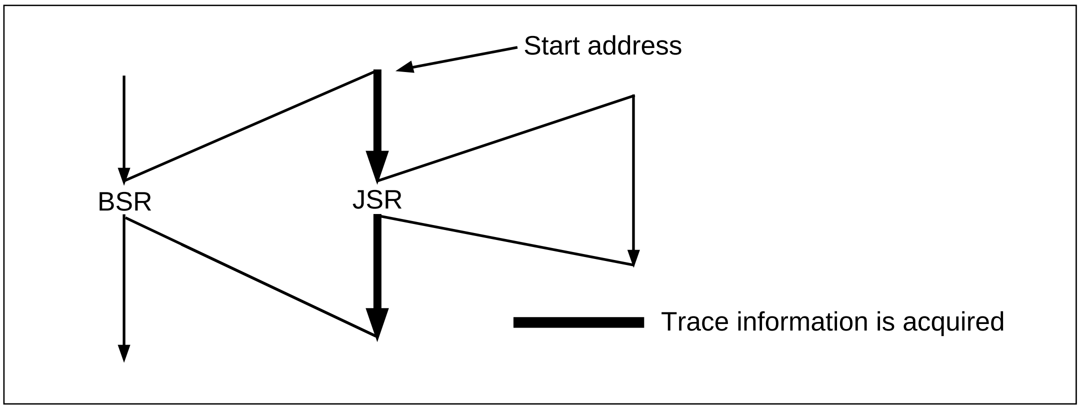


Figure 1-24 Subroutine Display

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

Subroutine Step Execution: When executing a JSR or BSR 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 internal ROM, internal RAM, user RAM, or 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, a specified address must be at the start of an instruction.

1.7 Execution Time Measurement

1.7.1 Execution Time Measurement

The user can measure the user program execution time by specifying with the GO command. This function has three modes.

- Normal mode
- Time interval measurement mode 1
- Time interval measurement mode 2

Normal Mode: In this mode, the emulator measures the total execution time from when the user program is started with the GO command to when it is stopped by a break. Normal mode is selected as a default when no option is specified with the GO command. The maximum time period that can be measured is 305 hours.

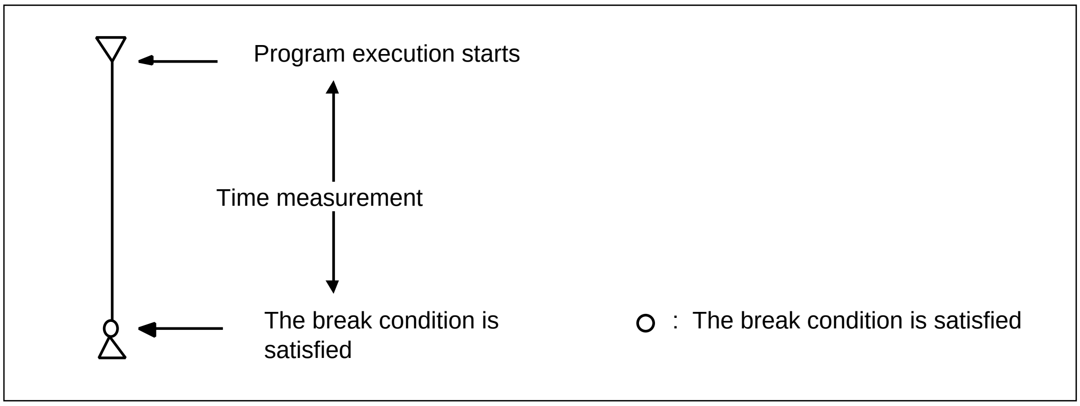

Program execution starts
Time measurement
The break condition is satisfied
○ : The break condition is satisfied

Figure 1-25 Normal Mode Time Measurement Range

Time Interval Measurement Mode 1: The emulator measures the elapsed time between the satisfaction of break conditions 2 and 1, up to a maximum of 305 hours. Set the break conditions, using BREAK_CONDITION1,2 commands. Time interval measurement mode 1 is specified using GO command option I1.

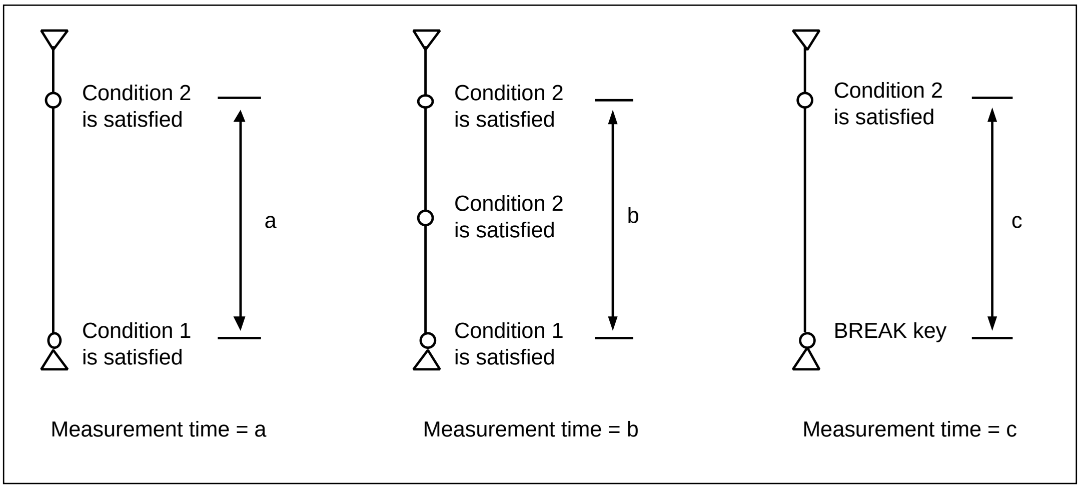


Figure 1-26 Time Interval Measurement Mode 1

Even if break condition 2 is satisfied many times before break condition 1, the emulator measures the time from the first occasion on which break condition 2 is satisfied. When this mode is specified, break conditions specified by the BREAK_CONDITION3,4,5,6 and the software break condition are invalid.

Time Interval Measurement Mode 2: In this mode, the time intervals between the satisfaction of break condition 2 (BREAK_CONDITION2) and break condition 1 (BREAK_CONDITION1) are added together (up to a maximum of 305 hours). Without stopping program execution, this mode is selected by specifying option "I2" with the GO command.

In time interval measurement mode 1, a break occurs after the break condition 2 and then break condition 1 are satisfied. However, in this mode, even if break condition 1 is satisfied, a break does not occur. When this mode is specified, software break is invalidated and the program can only be stopped using the (BREAK) key or (CTRL) + C keys.

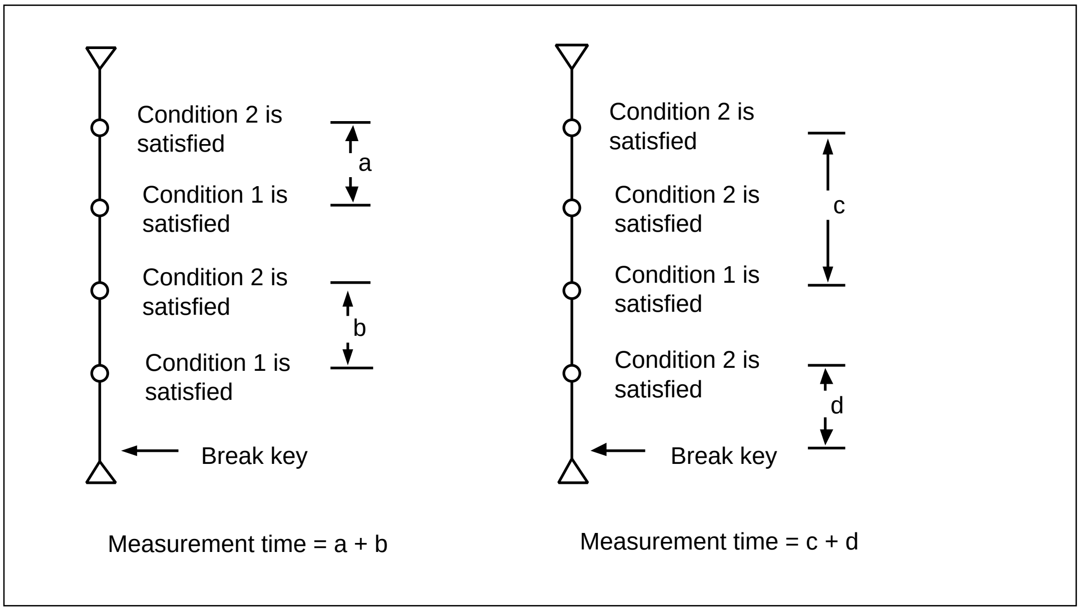


Figure 1-27 Time Interval Measurement Mode 2

1.7.2 Subroutine Time Measurement and Number of Times Measurement

The subroutine time and number of times the subroutines are executed can be measured based on the total program execution time by the PERFORMANCE_ANALYSIS command. Specify the subroutine to be measured with start and end addresses. A maximum of three subroutines can be measured at once. Subroutine time measurement starts when the program within the range from the start address to the end address is prefetched, and halts when a program not within the range is prefetched. The number of times is incremented when the end address of a subroutine is prefetched. For details on the PERFORMANCE_ANALYSIS command, refer to section 7.2.32, PERFORMANCE_ANALYSIS.

The measurement results are displayed in the following three ways:

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

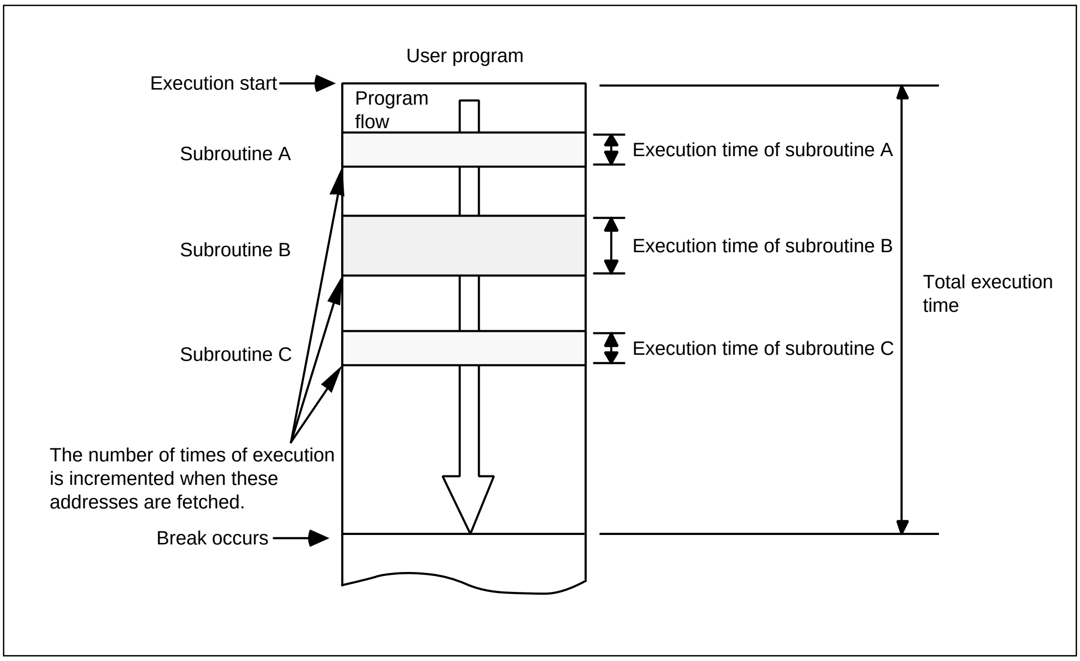

User program
Execution start
Program flow
Subroutine A
Execution time of subroutine A
Subroutine B
Execution time of subroutine B
Total execution time
Subroutine C
Execution time of subroutine C
The number of times of execution is incremented when these addresses are fetched.
Break occurs

Figure 1-28 Subroutine Time Measurement and Number of Times Measurement

1.8 Trigger Output

During user program execution, the emulator outputs a low-level pulse from the trigger output probe under the following three conditions. When using this pulse as an oscilloscope trigger input signal, it becomes easy to adjust the user system hardware.

- Cycle reset
- Trace condition satisfaction (When trigger output is specified)
- Hardware break condition satisfaction

In the following state, the emulator always outputs a low-level signal from the trigger output probe:

- Command input wait state

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 MCU. Cycle reset is specified with the GO command.

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 probe at the end of the bus cycle during which the hardware break condition is satisfied. The EXECUTION_MODE command can be used here to specify whether a break is to occur when the hardware break condition is satisfied. When the hardware break condition is satisfied with break specified, the emulator enters the command input wait state, and the emulator begins to output a continuous low-level signal. Specify the hardware break condition and the emulation mode with the BREAK_CONDITION1 to 6 and the GO command, respectively. When a sequential break condition is specified in GO command emulation mode, the emulator outputs a low-level pulse when each condition is satisfied.

Figure 1-29 shows the output timing.

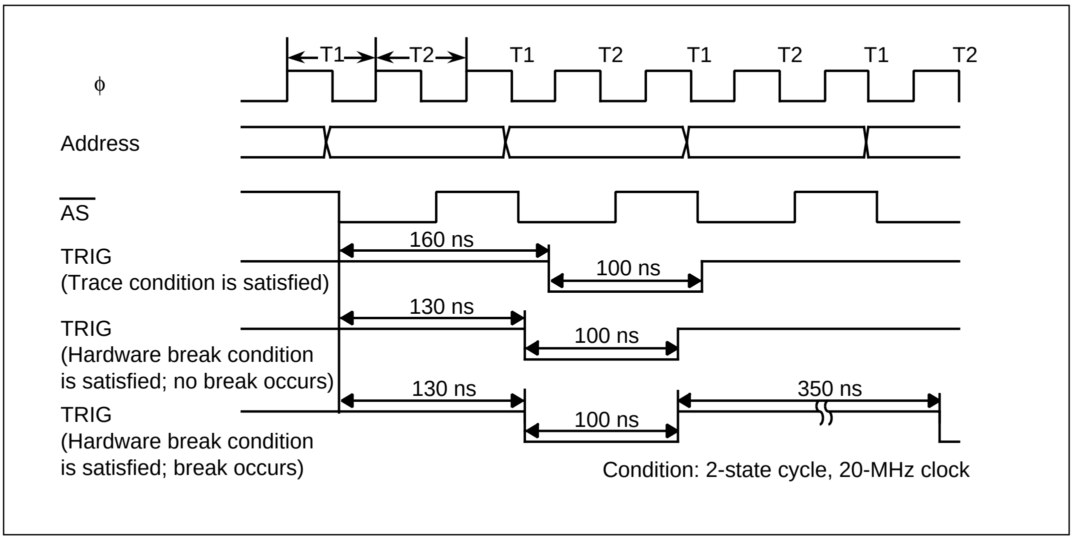

T1
T2
T1
T2
T1
T2
T1
T2
ϕ
Address
AS
TRIG
(Trace condition is satisfied)
160 ns
100 ns
TRIG
(Hardware break condition
is satisfied; no break occurs)
130 ns
100 ns
TRIG
(Hardware break condition
is satisfied; break occurs)
130 ns
100 ns
350 ns
Condition: 2-state cycle, 20-MHz clock

Figure 1-29 Pulse Output Timing

Note: The pulse output timing and pulse width depends on the specified condition, as shown in figure 1-29.

Command Input Wait State: The emulator always outputs a low-level signal from the trigger output probe in command input wait state.

1.9 Memory Access Function

Memory Management: The MCU can handle a maximum of 16 Mbytes of actual memory space. The emulator manages this space in 128-kbyte or 1-Mbyte units with the following memory attributes.

- User memory (User system memory)
- Emulation memory (512-kbyte standard memory and 1-Mbyte or 4-Mbyte optional memory)
- Write-protected memory
- Access-prohibited memory (guarded memory)

For details, refer to section 3, MCU Function Support, and section 7.2.27, MAP.

Memory Display, Modification, and Transfer: Memory display, modification, and transfer are done, using the following commands.

ASSEMBLE	DATA_CHANGE	DATA_SEARCH
DISASSEMBLE	DUMP	FILL
MEMORY	MOVE	MOVE_TO_RAM
LOAD	SAVE	
FILE_LOAD	FILE_SAVE	
LAN_LOAD	LAN_SAVE	
INTFC_LOAD	INTFC_SAVE	

1.10 MCU Control and Status Check

The emulator is capable of switching the clock signal supplied to the MCU, 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), the internal clock (8 MHz or 20 MHz), or the crystal oscillator installed on the emulator pod. To switch the clock, refer to section 7.2.10, CLOCK, and note the following.

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

  — When an external clock is supplied from the user system, selects the user clock
  — When a crystal oscillator is installed to the emulator pod, selects the crystal oscillator
  — Selects the internal clock (8 MHz)

Substitution of Internal ROM and RAM: The MCU has internal ROM/RAM. The emulator is forcibly prevented from accessing MCU internal ROM/RAM. These are substituted by off-chip RAM. The substitute RAM can be accessed in MCU bus cycles. Therefore, it does not affect realtime emulation.

Note: In the MCU extended mode, some areas in memory space are assigned to the internal ROM/external address area/reserved area or the internal ROM/reserved area. In the actual MCU, these areas are determined to be the external address area or the reserved area after reset because the initial value of the EAE bit in internal I/O register BCRL is 1. In the emulator, when the ROM size is set to 128 kbytes with the MODE command, the emulator handles these areas as internal ROM if they are accessed with emulator commands. Therefore, to use these areas as internal ROM, be sure to clear the EAE bit to 0 by user program. When the internal ROM/external address area/reserved area is used as an external address area, parallel access cannot be made.

Strobe Signal Output at Emulation Memory Access: When emulation memory is accessed, the emulator outputs the address but does not output the strobe signals (AS, RD, HWR, and LWR signals). When the user system needs the strobe signals for the bus cycle that accesses memory, specify the strobe signal output with the EXECUTION_MODE command. In this case, the user WAIT signal input is also enabled. For details, refer to section 7.2.17, EXECUTION_MODE.

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

RES, STBY, BREQ, WAIT, IRQ0 to IRQ7, and NMI

The CHECK command can check the same signals that are checked at system initiation.

Note: The IRQ0 to IRQ7 are checked only in the H8S/2655 series; they are not checked in the H8S/2245 series.

Emulator Execution Status Display: The emulator can display the execution status information listed in table 1-6. To display the execution status, use the STATUS command.

Table 1-6 Execution Status Display

Operation Status
Radix type
Number of breakpoints specified with the BREAK command
BREAK_CONDITION1 to 6 command specification
TRACE_CONDITION command specification
Host computer interface conditions
Number of defined symbols
Number of defined line number symbols
Information displayed with the STEP_INFORMATION command during STEP and STEP_OVER command execution
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
Coverage function display
Remaining emulation memory

1.11 Emulation Monitoring Function

The emulator monitors the emulation status such as memory accesses or user program execution. Two kinds of status are monitored.

- MCU operating status
- User system power and clock status

MCU Operating Status: When executing the program with the GO command, the emulator monitors the 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. Table 1-7 shows the execution status to be 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 7.2.21, GO.

Table 1-7 Operating Status Display

Display	Meaning
** PC=xxxxxxxx [yyyyyy = zzzz] (a) (b) (c)	
(a) Program fetch address	Current program fetch address is displayed.
(b) Memory address (c) Memory contents	When specifying option TM with the GO command, the specified memory address and its contents are displayed.
** VCC DOWN	User system Vcc (power voltage) is off. The user program stops.
** RESET	RES signal is low. The MCU has been reset.
** WAIT A = xxxxxxxx xxxxxxxx: Address bus value	WAIT signal is low. The address bus value is displayed.
** HARDWARE STANDBY	STBY signal is low. However, because STBY signal is not input to the MCU, the MCU is not in hardware standby state and is operating. Only this message is displayed.
** SOFTWARE STANDBY	MCU is in software standby state.
** SLEEP	MCU is in sleep state.
** BREQ	BREQ signal is low.
** BACK	BACK signal is low.
** TOUT A=xxxxxxxx xxxxxxxx: Address bus value	Bus cycle has stopped for 1.28 ms or more. The address bus value is displayed.

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

Table 1-8 Operation at Power-Off

Operation Status		Emulator Operation
At initiation	User system power is off (XTAL and user clock installed)	Waits for user system power to be turned on, after which the emulator is initiated with the user clock.
	User system power is off (XTAL not installed, user clock installed)	
	User system power is off (XTAL installed, user clock not installed)	Waits for user system power to be turned on, after which the emulator is initiated with the user clock.
After initiation	Command input wait state	Displays *** VCC DOWN and enters command input wait state. After the user system power is turned on, RESET IN BY E7000! is displayed. The emulator is reset and re-enters command input wait state (no change in clock source).
	During user program execution and while in parallel mode	Displays *** VCC DOWN and stops GO command execution. After the user system power is turned on, RESET IN BY E7000! is displayed. The emulator is reset and enters command input wait state (no change in clock source).

Notes: 1. User-system power-off is detected at about 2.0 V.

2. When the user system interface cable is connected to the emulator, power must be supplied from the cable head. Thus, when using the emulator on its own (stand-alone mode), be sure to disconnect the user system interface cable.

1.12 Setting Operating Voltage and Operating Frequency

The emulator executes user programs within the operating voltage range from 2.7 V to 5.5 V supplied from the user system. Specify the combination of the operating voltage and the operating frequency according to figure 1-30.

Note: The emulator monitors the operating voltage from the user system and whether the user system interface cable is connected. When using the emulator on its own (stand-alone mode), be sure to disconnect the user system interface cable from the emulator pod connector.

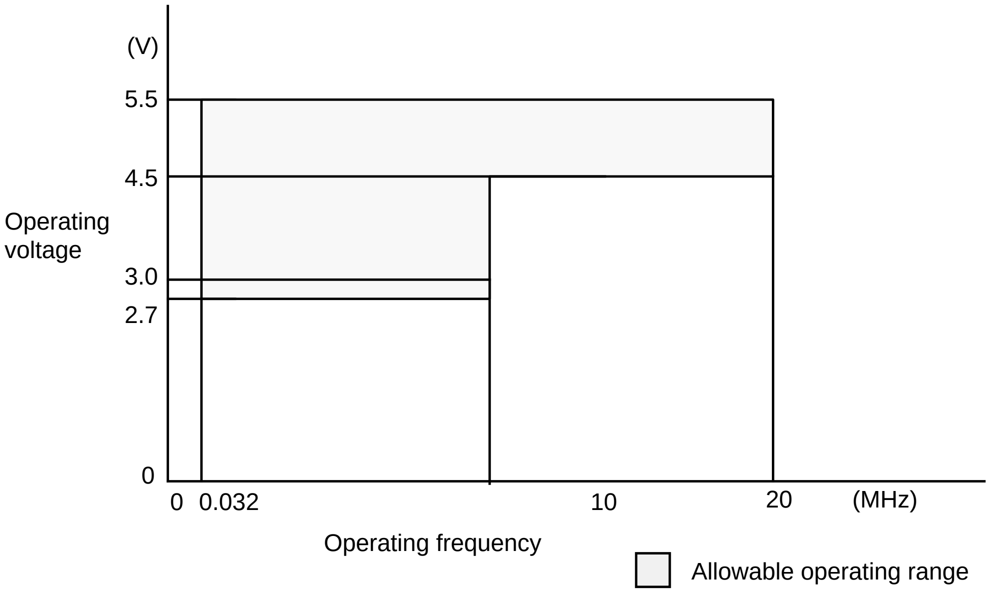


Figure 1-30 Relation between Operating Voltage and Operating Frequency

1.13 Symbolic Debugging

The emulator has a symbolic debugging function which uses the source program symbols (variables and line numbers). This function is explained below.

1.13.1 Symbol Definition

Definition Methods: There are three ways to define symbols.

- Load of load module with debugging information
  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 defined as label.

- ASSEMBLE command
  Label names are defined using the ASSEMBLE command.

Symbol Attributes: Table 1-9 lists the attributes of the symbols which may be defined within the load module (SYSROF type) debugging information.

Table 1-9 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.

1.13.2 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
```

1.13.3 Symbol Deletion

The symbol deletion function of the SYMBOL command deletes all symbols at the same time. Specific symbols cannot be deleted.

1.13.4 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 nesting 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.

1.14 Assembly Function

1.14.1 Overview

The ASSEMBLE command enables line assembly as shown in figure 1-31.

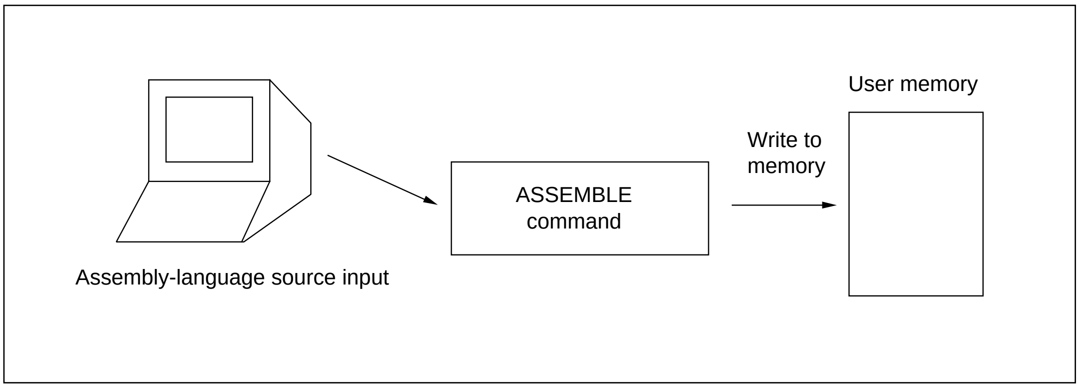


Figure 1-31 Assembly Function

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

1.14.2 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 a symbol name can be used, provided that it is an upper-case character. The label name must start in the first column.

<instruction mnemonic>: Any instruction mnemonic described in the H8S/2600 Series, H8S/2000 Series Programming Manual and any assembler directive listed in table 1-10 can be used.

<operand>: Any mnemonic described in the H8S/2600 Series, H8S/2000 Series Programming Manual can be used (table 1-11).

<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 1-10 Assembler Directives

Directive	Operand	Description
Δ.DATA[.s]Δ	<value>[,<value>...]	• Reserves an area for initialized fixed-length data. The size of the area is equal to the unit length given by s: B (byte), W (word) or L (long word). Default size is B. • 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 B. • Up to 16-Mbyte area can be reserved at one time.

Table 1-11 Operand Description

Format	Addressing Mode	Remarks
ERn	Register direct	ERn: 32-bit register
En		En: 16-bit register
Rn		Rn: 16-bit register
RnH		RnH: 8-bit register
RnL		RnL: 8-bit register
@ERn	Register indirect	ERn: Register name
@(d:16, ERn)	Register indirect with displacement	d: Displacement value
@(d:32, ERn)		ERn: Register name
@ERn+	Register indirect with post-increment	ERn: Register name
@-ERn	Register indirect with pre-decrement	ERn: Register name
@aa:8	Absolute address	aa: Absolute address value
@aa:16		
@aa:24		
@aa:32		
#xx:8	Immediate data	xx: Immediate data value
#xx:16		
#xx:32		
@(d:8, PC)	PC relative with	d: Displacement value
#(d:16, PC)		PC: Program counter value
@@aa:8	Absolute address	aa: Absolute address value

Notes: 1. For the address value, immediate data value, and displacement value, symbols or expressions (addition or subtraction) can be used. However, only address values can be displayed at disassembly.

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

1.14.3 Definition of Label Names as Symbols

The names of labels in the assembly-language code are all defined as symbols, which may be used in each command. In line assembly, a symbol name cannot be defined if it has already been defined.

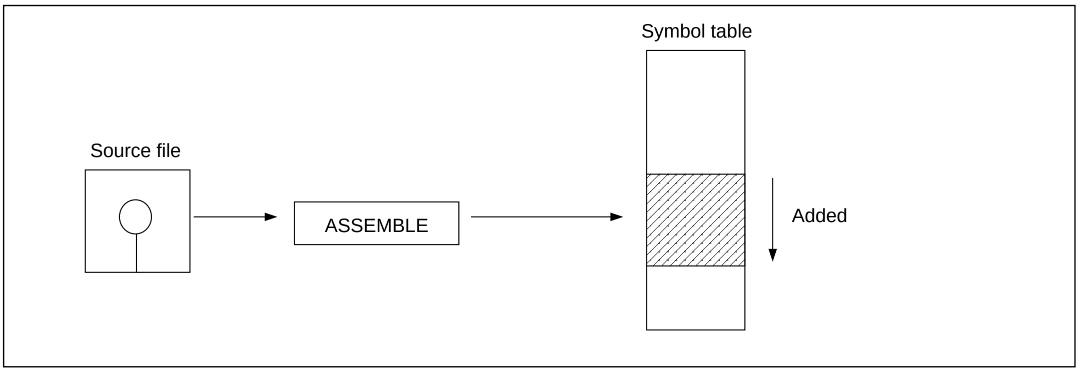


Figure 1-32 Label Name Definition

The maximum number of labels is 1,024. However, if symbols have been already defined, the maximum number is smaller.

1.14.4 Label Name Reference

A label name can be referenced by specifying it in an operand field in an assembly-language source statement using the following addressing modes:

- PC relative addressing mode: For instructions BRA, BRN, Bcc, and BSR
- Absolute addressing mode: For instructions other than the above
- 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. However, 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.

1.14.5 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 7.2.15, DISASSEMBLE.

Section 2 Differences between the Actual MCU and the Emulator

2.1 Register Values after Reset

When the emulator system is initiated, or when the emulator resets the MCU as a result of a command, such as the CLOCK command switching the clock or the RESET command, the general-purpose registers and part of the control registers are initialized. Note especially that the ER7 value is initialized as shown in table 2-1.

Table 2-1 Differences between Initial Values of the Actual MCU and Emulator Registers

Status	Register	Emulator	Actual MCU
Emulator initiation (power-on) or reset with command	PC	Vector address	Vector address
	CCR	10000000	1XXXXXXX
	ER0 to ER6	00000000	Undefined
	ER7	Start address of internal I/O – 2	Undefined
	EXR	00000111	Undefined
	MACH	00000000	Undefined
	MACL	00000000	Undefined

Note: X is an undefined value.

2.2 User Interface

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

2.3 Crystal Oscillator

The user can install a crystal oscillator into the emulator pod. When using the crystal oscillator, note that it must have a frequency from 10 MHz to 20 MHz. When using frequencies outside this range, input the external clock from the EXTAL pin.

2.4 Load Capacitance

Loads applied to the emulator and user system interface cable are larger than those in the actual MCU. Therefore, buffers must be connected to the user system interface for large capacitance signals on the user system.

2.5 A/D and D/A Converters

Because the user system interface cable, printed circuit boards, and protective circuits are connected between the MCU in the emulator pod and the user system, the conversion precision is lower than that of the actual MCU. At final debugging of the user system using the A/D or D/A converter, use the actual MCU (ZTAT microcomputer chip).

2.6 Write Data Buffer Function

In the actual MCU, the write data buffer function can be used. The write data buffer function is valid when an external data cycle is performed. However, when the emulation memory is used, the emulator uses the internal data bus and the write data buffer function is invalidated in the emulator. Therefore, when the write data buffer function is enabled, program execution time using the emulation memory is different from that using the user memory.

2.7 Internal ROM/External Address Area/Reserved Area

In the MCU extended mode, some areas in memory space are assigned to the internal ROM/external address area/reserved area or the internal ROM/reserved area. In the actual MCU, these areas are determined to be the external address area or the reserved area after reset because the initial value of the EAE bit in internal I/O register BCRL is 1. In the emulator, when the ROM size is set to 128 kbytes with the MODE command, the emulator handles these areas as internal ROM if they are accessed with emulator commands. Therefore, to use these areas as internal ROM, be sure to clear the EAE bit to 0 by user program. When the internal ROM/external address area/reserved area is used as an external address area, parallel access cannot be made.

Section 3   MCU Function Support

The MCU has seven operating modes. These operating modes are supported by the emulator as described below.

3.1 Operating Mode Setting

The MCU settings such as MCU mode, operating mode, number of pins, and peripheral functions can be specified using the MODE command. For details on MCU settings, refer to section 4, Operating Examples in Part I, E7000 Guide and in Part II, E7000PC Guide. The following describes how to specify the operating mode.

Each MCU operating mode can be set by mode setting pins (MD0 to MD2) as shown in table 3-1. In the emulator, two methods for specifying the operating mode are used: one is the E7000 mode in which the desired operating mode can be set regardless of the setting of the mode setting pins on the user system, and the other is the user mode in which the operating mode is determined by the setting of the mode setting pins on the user system. For details, refer to section 7.2.29, MODE. When the user system is not connected, the E7000 mode is automatically selected. The emulator can also read the mode setting pin status of the user system in E7000 mode without affecting the emulator MCU operating mode.

Table 3-1   MCU Operating Mode Selection

Operating Mode	MD2	MD1	MD0	H8S/2655 Series and H8S/2245 Series
Mode 0	0	0	0	—
Mode 1	0	0	1	Normal extended mode without internal ROM
Mode 2	0	1	0	Normal extended mode with internal ROM
Mode 3	0	1	1	Normal single-chip mode
Mode 4	1	0	0	Advanced extended mode without internal ROM (16 bits)
Mode 5	1	0	1	Advanced extended mode without internal ROM (8 bits)
Mode 6	1	1	0	Advanced extended mode with internal ROM
Mode 7	1	1	1	Advanced single-chip mode

The MCU mode, operating mode, number of pins, and peripheral functions previously set can be saved to the configuration file in the emulator system disk. When initializing with this disk, the emulator initiates the system with the settings specified with the MODE command. When the operating mode is set with the MODE command, the emulator system program terminates and must be restarted.

3.2 Memory Space

The MCU has a maximum memory space of 16 Mbytes, within which memory can be set in units of 128 kbytes (in units of 1 Mbyte on an optional emulation memory board).

U:	User system memory
S:	Standard emulation memory
E:	Optional emulation memory

A write-protected area and an access-prohibited (guarded memory) area can be each allocated to memory in minimum units of 128 kbytes.

W:	Write-protected
G:	Access-prohibited

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

3.2.1 Internal ROM Area

The emulator includes substitute RAM for the MCU internal ROM. In operating modes including internal ROM, the substitute RAM is accessed if an attempt is made to access the internal ROM, regardless of the MAP command attribute settings.

In addition, the internal ROM area access differs between user program execution and the emulator commands.

• Access by user program execution:	Read only, write disabled (program is terminated if attempted.)
• Access with an emulator command:	Read/write enabled

Therefore, the internal ROM contents can be changed or an object program can be loaded using commands such as MEMORY and LOAD, but they cannot be rewritten from the user program. If this is attempted, the user program is terminated.

The internal ROM area is accessed in one state.

Note: In the MCU extended mode, some areas in memory space are assigned to the internal ROM/external address area/reserved area or the internal ROM/reserved area. In the actual MCU, these areas are determined to be the external address area or the reserved area after reset because the initial value of the EAE bit in internal I/O register BCRL is 1. In the emulator, when the ROM size is set to 128 kbytes with the MODE command, the emulator handles these areas as internal ROM if they are accessed with emulator commands. Therefore, to use these areas as internal ROM, be sure to clear the EAE bit to 0 by user program. When the internal ROM/external address area/reserved area is used as an external address area, parallel access cannot be made.

3.2.2 Internal RAM Area

The emulator includes substitute RAM for the MCU internal RAM. When the internal RAM is enabled, if an attempt is made to access addresses H'F800 to H'FBFF (H'FFF800 to H'FFFBFF in advanced mode) in the internal RAM, the internal RAM is accessed, and if an attempt is made to access the other areas of the internal RAM, this substitute RAM is accessed regardless of the memory attribute set with the MAP command. In addition, the user program is not terminated even if the internal RAM area is written to or accessed by the user program while the attribute setting is write-protected (attribute W) or access prohibited (attribute G). To break the user program when the internal RAM area is written to or accessed, use the BREAK_CONDITION1,2,3,4 commands.

The internal RAM area can be accessed by the user program and with the emulator commands, and can be accessed in one state.

If the internal RAM is disabled, it is used as an external area.

3.2.3 Internal I/O Area

If an attempt is made to access the internal I/O area, the internal I/O area in the MCU installed in the emulator is accessed regardless of the memory attribute set with the MAP command. In addition, the user program is not terminated even if the internal I/O area is written to or accessed by the user program while the attribute setting is write-protected (attribute W) or access prohibited (attribute G). To break the user program when the internal I/O area is written to or accessed, use the BREAK_CONDITION1,2,3,4 commands.

The internal I/O area can be read from or written to by the user program or with emulator commands. When writing to the internal I/O area with an emulator command (MEMORY command), the following warning message is displayed and the emulator starts writing without verifying.

```
*** 86:  INTERNAL I/O AREA
```

3.2.4 Unusable Area (Only in Single-Chip Mode)

In the emulator, emulation memory can be allocated to the unusable area in single-chip normal mode and single-chip advanced mode. In this case, this area can be read from or written to by the user program or with emulator commands. If the emulation memory is write-protected, the user program is terminated if this area is written to by the user program.

Such areas can be used as work memory for debugging the user program, but they cannot be accessed in the actual MCU. After debugging is completed, change the program so as not to access unusable areas.

3.2.5 External Memory Area

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

The emulator controls memory areas in memory block units as shown in figure 3-1. The user system memory and the emulation memory (standard and optional emulation memory) in the emulator cannot be allocated in the same block; the same type of memory can be allocated in the same block.

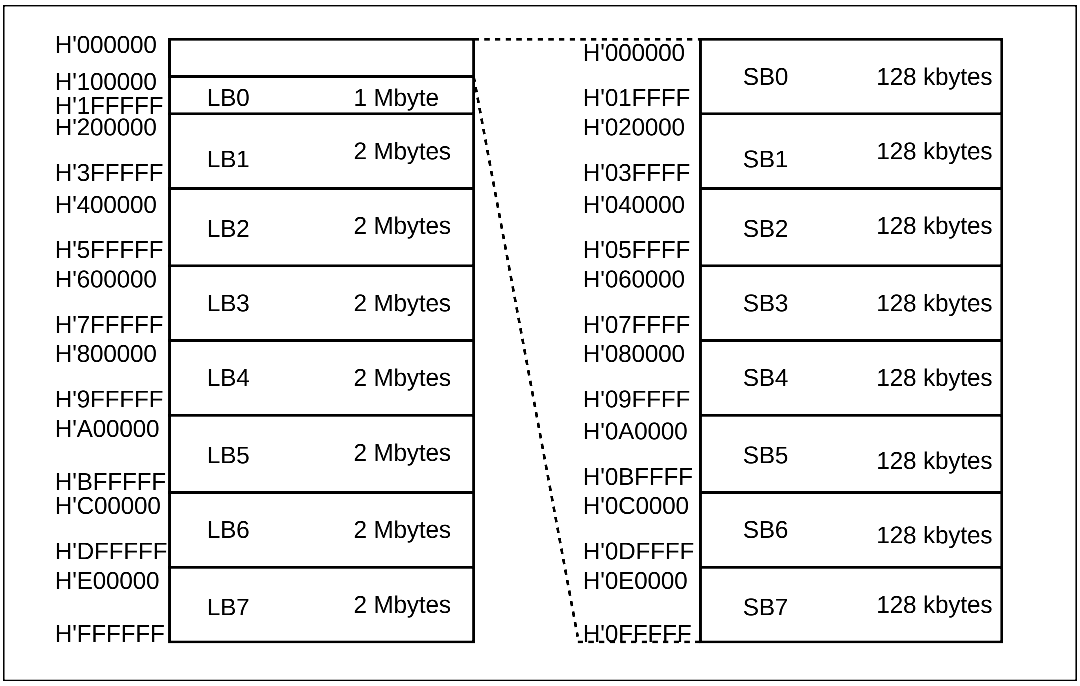


Figure 3-1 Memory Blocks

Note: In the MCU extended mode, some areas in memory space are assigned to the internal ROM/external address area/reserved area or the internal ROM/reserved area. When the ROM size is set to 128 kbytes with the MODE command, the emulator handles these areas as internal ROM if accessed with emulator commands. Therefore, in this state, these areas cannot be used as external address areas. To use the areas as external address areas, set the ROM size to 64 kbytes with the MODE command. When these areas are used as external address areas, parallel access cannot be made.

3.2.6 Reserved Areas

When using the emulator, emulation memory can be allocated to reserved areas, which can be accessed by either the user program or emulator commands. If the emulation memory attribute is set to write-protected, however, user program execution stops when this area is written to by the user program.

Note, however, that the actual MCU does not have such reserved areas. Therefore, these areas can be used only as work memory for debugging the user program, but they cannot be accessed in the actual MCU. After debugging is completed, change the program so as not to access reserved areas.

Reserved areas are accessed in the same way as external memory areas.

Note: In the MCU extended mode, some areas in memory space are assigned to the internal ROM/external address area/reserved area or the internal ROM/reserved area. When the ROM size is set to 128 kbytes with the MODE command, the emulator handles these areas as internal ROM if accessed with emulator commands. Therefore, in this state, these areas cannot be used as external address areas. To use the areas as external address areas, set the ROM size to 64 kbytes with the MODE command. When these areas are used as external address areas, parallel access cannot be made.

3.3 Low-Power Modes

For reduced power consumption, the MCU has middle speed, sleep, module stop, hardware standby, and software standby modes.

3.3.1 Hardware Standby Mode

The hardware standby mode is switched by the STBY signal input. However, since the STBY signal from the user system is not input to the MCU, the emulator does not support this mode.

The status of the STBY signal sent from the user system can be monitored.

3.3.2 Sleep and Software Standby Modes

These modes are switched by executing the SLEEP instruction, and are cleared by break condition matching (including break key input) as well as by normal clear sources, and program breaks.

Trace information is not acquired during these 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.**

3.3.3 Medium-Speed and Module Stop Modes

These modes are switched by the settings of the I/O registers in the MCU: the emulator does not control these modes.

3.4 Interrupts

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

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

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

Notes:

1. **The loop program specified by the BACKGROUND_INTERRUPT command must be stored in the internal RAM area.**
2. **In interrupt processing, hardware break, software break, and access to the write-protected area or guarded area are detected and a break occurs.**
3. **Information on interrupt processing cannot be traced.**

3.5 Control Input Signals (RES, WAIT, BREQ)

The MCU control input signals are RES, WAIT, and BREQ. The RES signal is valid only during emulation with the GO command. The WAIT and BREQ signals are valid during emulation with the GO, STEP, or STEP_OVER command. Therefore, while the emulator is waiting for command input, the user cannot input RES, WAIT or BREQ signals to the MCU. BREQ signals can be masked using the EXECUTION_MODE command.

3.6 Watchdog Timer (WDT)

The WDT only operates during emulation (by the GO, STEP, or STEP_OVER command), and does not operate when the emulator is waiting for command input. The timer stops at a break and resumes when emulation restarts.

3.7 16-Bit Timer Pulse Unit (TPU) and 8-Bit Timer

The TPU and 8-bit timer operate during the command input wait state as well as during emulation. Even if the user program has stopped when a break condition is satisfied after the user program has been started with the GO command, the TPU and 8-bit timer continue 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 (SCI)

The serial communications interface signals are connected to the user system directly from the MCU on the emulator pod. Therefore, the interface is valid during the emulator command input wait state as well as during emulation. For example, when writing data to the transmit data register (TDR) with the MEMORY command, data is output to the TXD line after the serial communication interface output has been prepared.

3.9 Programmable Pulse Generator (PPG)

The PPG operates during the command input wait state as well as during emulation.

3.10 DMA Controller (DMAC)

The DMAC operates during the command input wait state as well as during emulation. When a transfer is requested, the DMAC executes a DMA transfer.

3.11 Data Transfer Controller (DTC)

The DTC operates during the command input wait state as well as during emulation. When a transfer is requested, the DTC executes a DTC transfer.

Note: The DTC is connected to the area from address H'F800 to H'FBFF (H'FFF800 to H'FFFBFF in advanced mode) in the internal RAM through a 32-bit bus. Therefore, for the transfers between the high-order 16 bits of the DTC and the internal RAM, trace information cannot be acquired and no break can be generated.

3.12 Bus Controller

3.12.1 Wait

The MCU wait state controller has a programmable wait mode and a WAIT pin input mode. The programmable wait mode is valid when the emulation memory or user external memory is accessed, but input to the user WAIT pin is only valid when user external memory is accessed. However, the EXECUTION_MODE command can be used to enable input to the user WAIT pin during emulation memory access cycles. When optional emulation memory is accessed, one wait state is always inserted to three states, and the input to the user WAIT pin is ignored. The input to the user WAIT pin is always enabled during refresh cycles.

3.12.2 Refresh

The refresh controller always operates in the emulator. Refresh cycles are executed even in the emulator command input wait state (trace information is not acquired).

3.12.3 Write Data Buffer Function

With the emulator, the write data buffer function can be used, except for access to the optional emulation memory. Use the standard emulation memory when using the write data buffer function.

3.12.4 BCRL EAE Bit

Some areas in MCU expanded mode are specified as an internal ROM/external address area/reserved area or an internal ROM/reserved area. To use these areas as internal ROM, the EAE bit of the BCRL, which is an internal I/O register of the MCU, must be set to 0.

The emulator cannot check the EAE bit value set by the user, and therefore, access to these areas with emulator commands is made to internal ROM. On the other hand, access with a user program can be made to the area specified by the EAE bit value: the external address area/reserved area when EAE is 1, and the internal ROM when EAE is 0. For example, a program is loaded into internal ROM but when the program is executed, the area specified by the EAE bit value is accessed. To use these areas as internal ROM during user program execution, the EAE bit must be cleared to 0 during reset processing by user program.

To use these areas as an external address area/reserved area, change the ROM size to 64 kbytes with the MODE command. In this case, parallel access cannot be performed.

3.13 I/O Port

The MCU I/O port can be used as peripheral module input/output pins or as an address/data bus. It is specified as I/O port pins according to the operating mode or internal register settings. The I/O port pins are also valid in the emulator command input wait state or during emulation.

The I/O port pins can be read from and written to by the MEMORY command.

3.14 A/D and D/A Converters

Analog I/O pins are directly connected to the user system from the MCU installed in the emulator. Therefore, they are valid in the emulator command input wait state as well as during emulation.

The A/D and D/A converters also have AVcc, AVss, Vref, and ADTRG pins. Because these converters operate with a special power supply, connect AVcc (power supply pin) and Vref (reference voltage pin) to the A/D and D/A conversion power supply and the reference power supply on the user system.

Notes:

1. **Even when not using the A/D and D/A converters, connect the AVcc and Vref pins to Vcc.**
2. **Because the user system interface cable, printed circuit boards, and protective circuits are connected between the MCU in the emulator pod and the user system, the conversion precision is lower than that of the actual MCU. At final debugging of the user system using the A/D or D/A converter, use the actual MCU (ZTAT microcomputer chip).**

Section 4 User System Interface

The emulator is connected to the user system with the user system interface cable. Probe signal trace and break can be enabled by connecting eight external probes to the user system.

User System Interface Circuits: The circuits that interface the MCU 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 for the signals that exceed the MCU AC timing specifications when the emulator is used. The other signals satisfy the MCU AC timing specifications.

The values in table 4-1 were obtained when using the emulator. These values are only reference values and not guaranteed values. Taking this in consideration, adjust the user system hardware

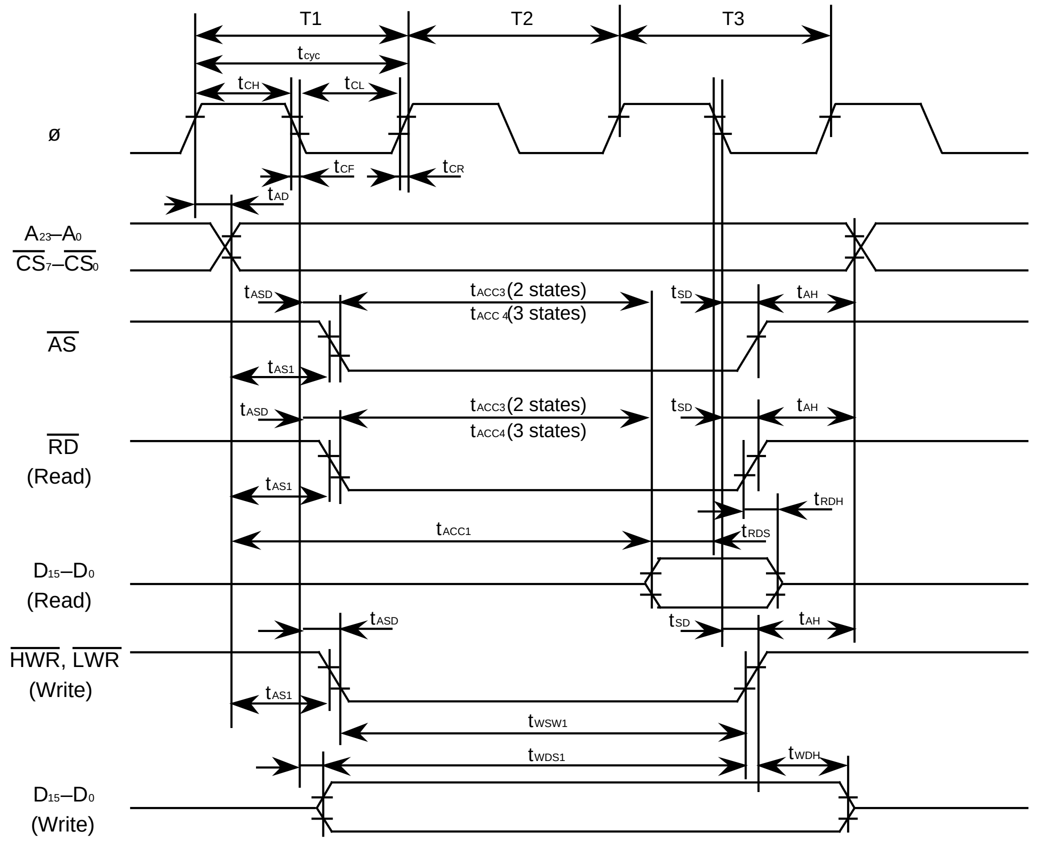
Table 4-1 Bus Timing

Parameter	MCU Specification (ns)	Emulator Specification (ns)
tRDS (5 V, 18-MHz operation)	15 (min)	39 (typ)
tACC1 (5 V, 18-MHz operation)	50 (max)	37 (typ)
tACC2 (5 V, 18-MHz operation)	105 (max)	93 (typ)
tACC3 (5 V, 18-MHz operation)	20 (max)	–5 (typ)
tACC4 (5 V, 18-MHz operation)	80 (max)	50 (typ)

Refresh Controller Bus Timing

Parameter	MCU Specification (ns)	Emulator Specification (ns)
tRAH (5 V, 18-MHz operation)	20 (min)	8 (typ)
tRAC (5 V, 18-MHz operation)	70 (max)	55 (typ)
tAA (5 V, 18-MHz operation)	45 (max)	39 (typ)
tCAC (5 V, 18-MHz operation)	25 (max)	16 (typ)
tWDS3 (5 V, 18-MHz operation)	40 (max)	20 (typ)

Figures 4-1 and 4-4 show the bus timing. Figure 4-5 shows the circuits that interface the emulator to the user system.



Shorten by one clock in 2-state access cycle because T3 state is deleted.

Figure 4-1 Basic Bus Cycle Timing in Extended Mode

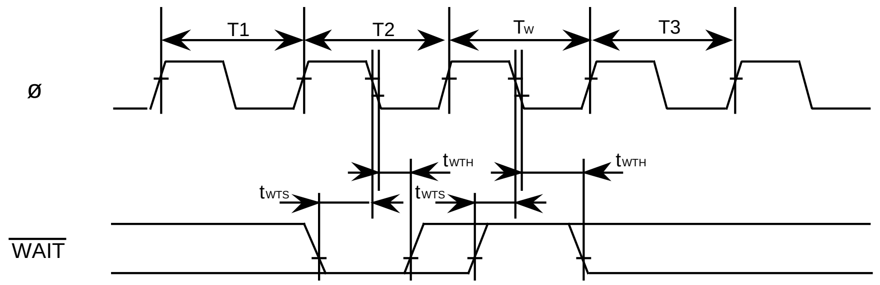


Figure 4-2 Bus Timing With One Wait State Inserted

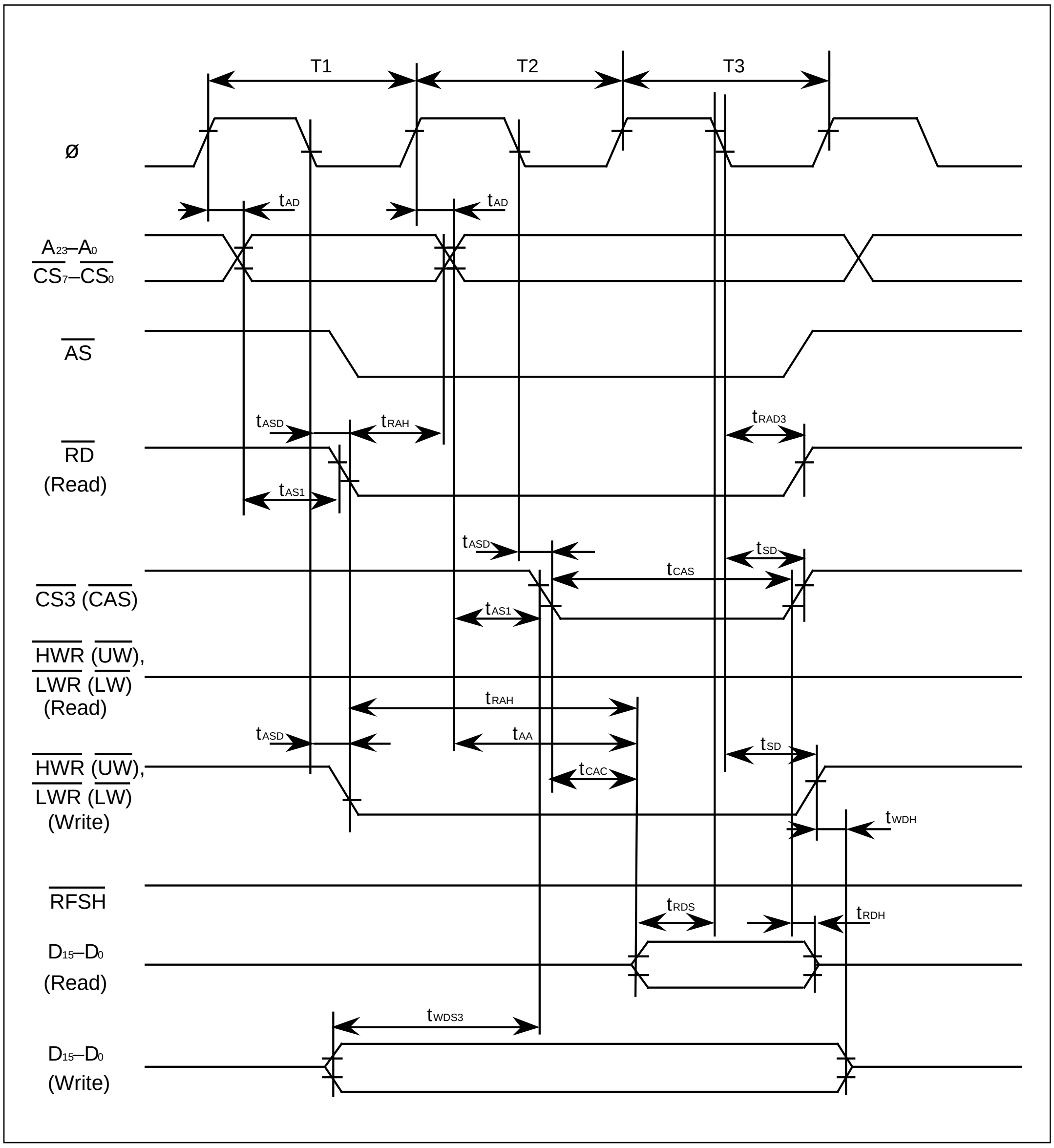


Figure 4-3 DRAM Read/Write Bus Timing (2WE Method)

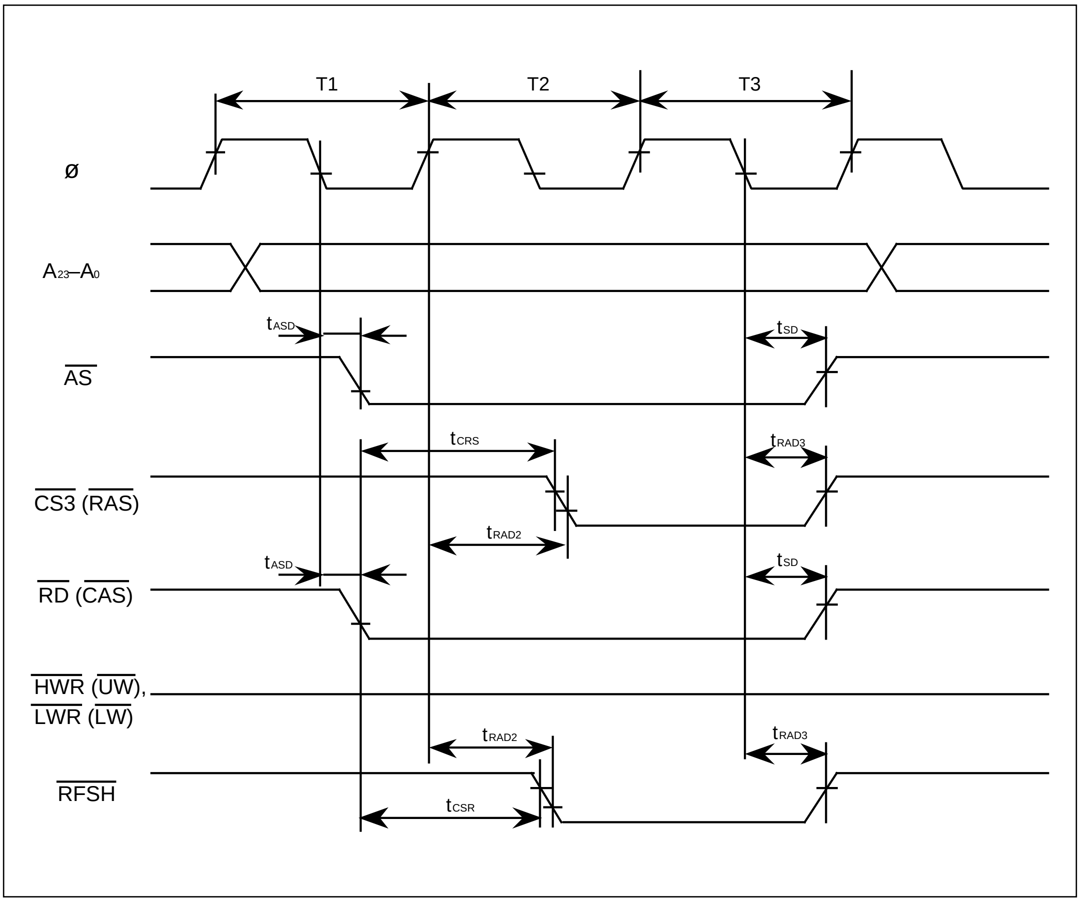
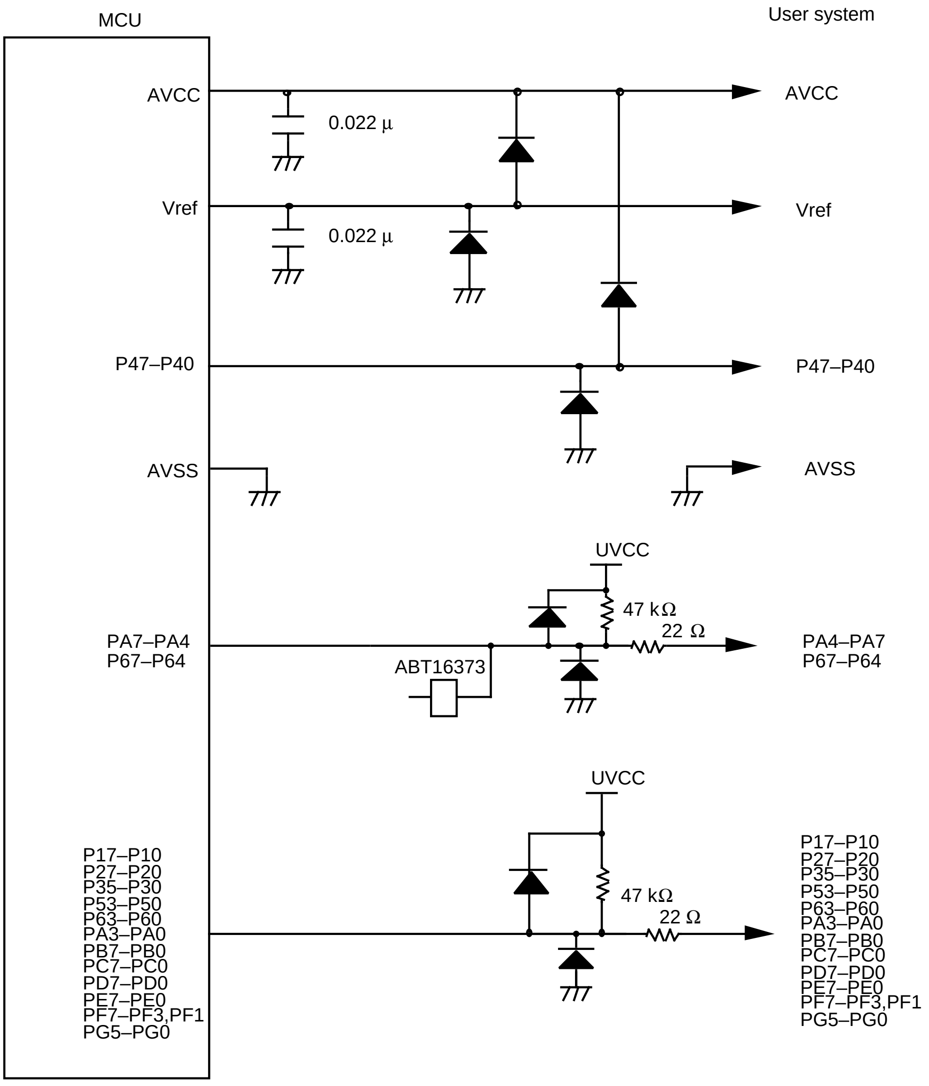


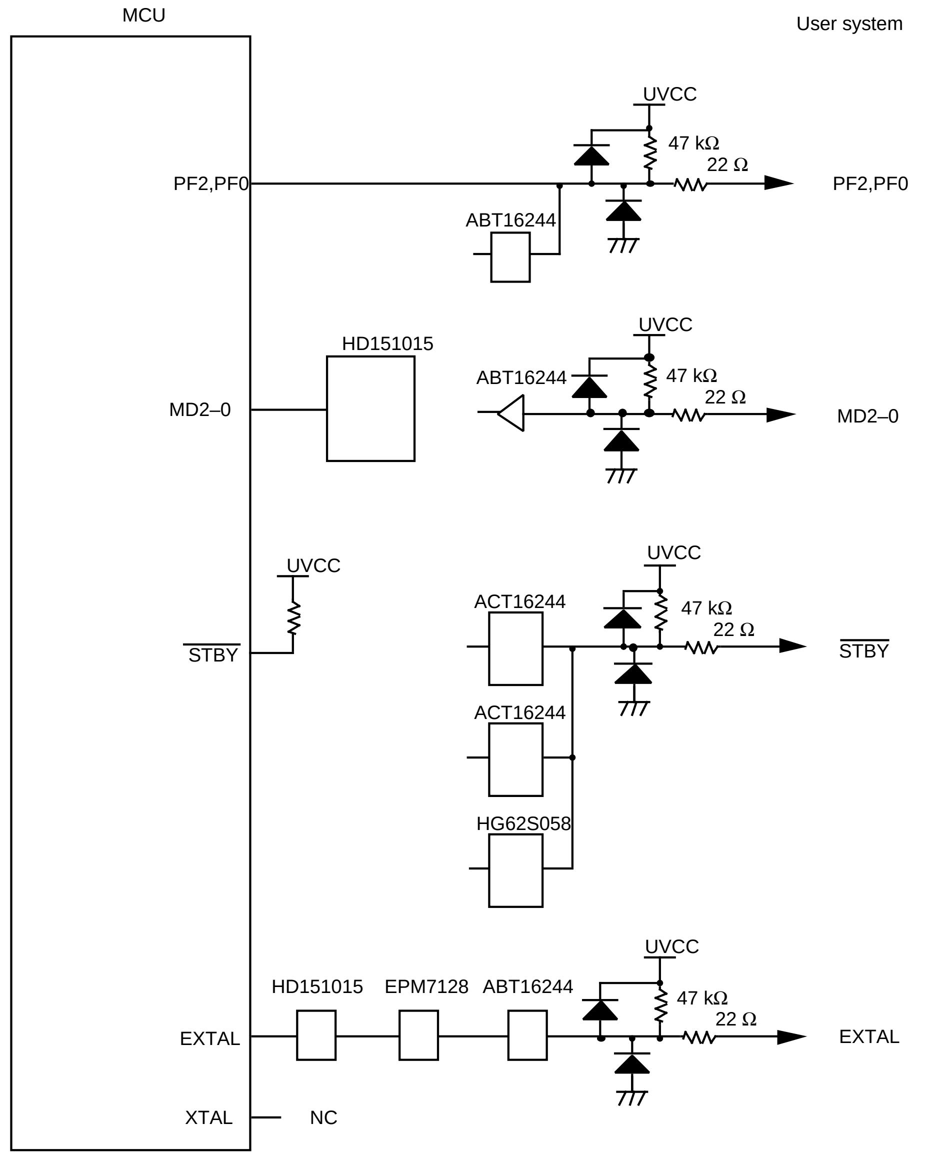
Figure 4-4 DRAM Refresh Cycle Bus Timing (2WE Method)



Note: UVCC is either of the following:

- User system VCC when the user system is connected via the user system interface cable
- 5-V emulator power when the user system interface cable is not connected

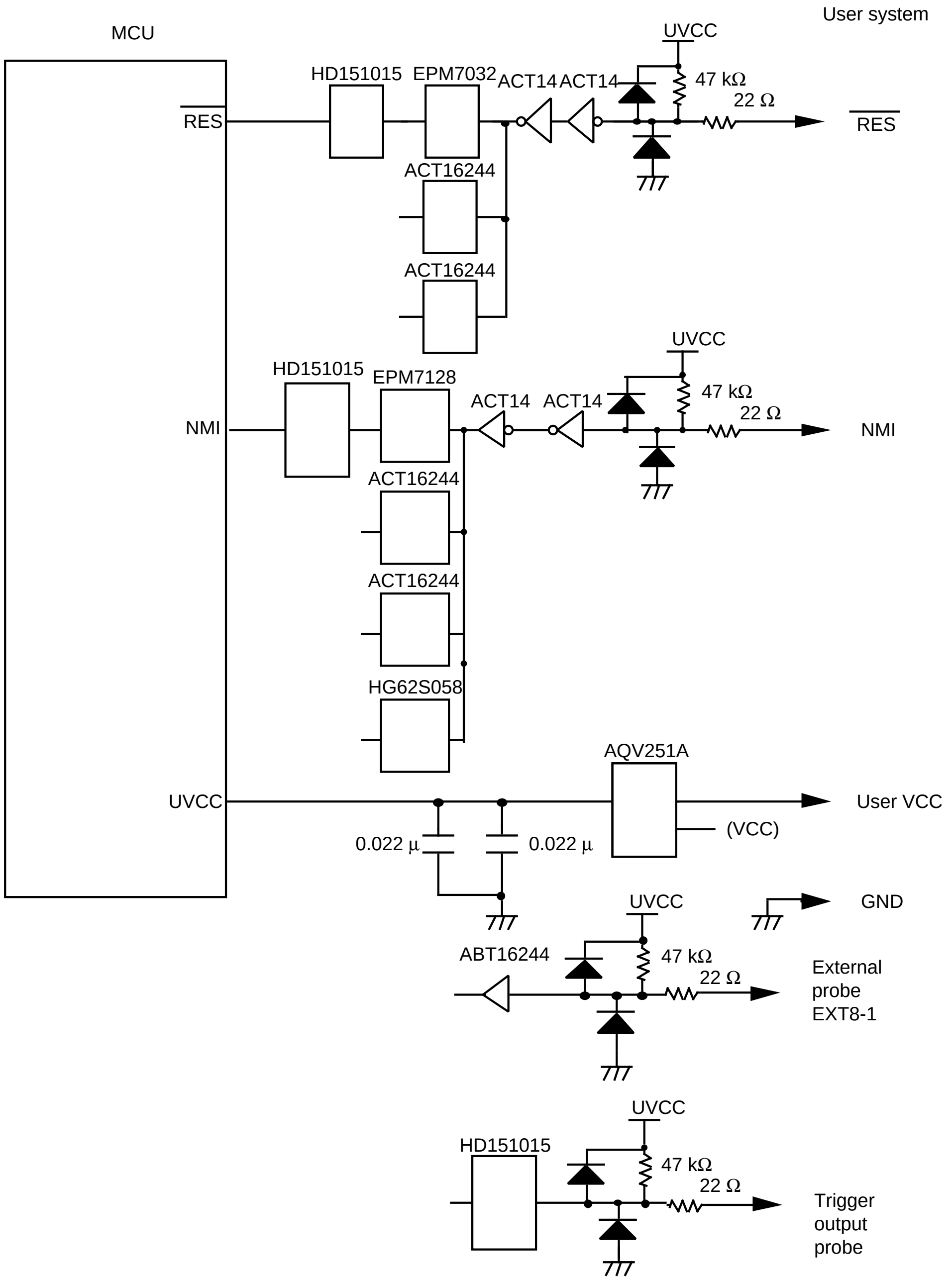
Figure 4-5 User System Interface Circuits



Notes: 1. UVCC is either of the following:
- User system VCC when the user system is connected via the user system interface cable
- 5-V emulator power when the user system interface cable is not connected

2. EPM7128 is an EPLD manufactured by Altera Corporation.
3. The emulator and the user system interface cable have load capacitance. If the load capacitance on the user system is larger, connect EXTAL through a buffer.

Figure 4-5 User System Interface Circuits (cont)



Notes: 1. UVCC is either of the following:
- User system VCC when the user system is connected via the user system interface cables
- 5-V emulator power when the user system interface cable is not connected

2. EPM7032 is an EPLD manufactured by Altera Corporation.
3. (VCC) is 5-V emulator power supplied from the E7000 or E7000PC.
4. User VCC is the user system power supply.

Figure 4-5 User System Interface Circuits (cont)

Section 5 Troubleshooting

The emulator internal system test checks the emulator's internal RAM and registers at power-on and at system program initiation.

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
 F : FORMAT FD                                (d)
 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=xxxxxxxx
  ```

  No data will be displayed if an error does not occur.

- 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

No data will be displayed if an error does not occur.

(c) E7000 system program is being loaded. If the E7000 system disk is not inserted, the E7000 monitor enters command input wait state.

(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 5.3, Troubleshooting Procedure, without loading the E7000 system program.

Internal System Test at E7000 System Program Initiation: The E7000 system program performs internal system tests, mainly on the E7000 registers, at its initiation.

```
**   E7000 SYSTEM LOADING   **

H8S/xxxx E7000 (HSxxxxEPD70SF) Vn.m
Copyright (C)  Hitachi, LTD. 199X                                      (a)
Licensed Material of Hitachi, Ltd.

LAN IP ADDRESS FILE LOADING                                            (b)
CONFIGURATION FILE LOADING                                             (c)
HARD WARE REGISTER  READ/WRITE CHECK                                   (d)
POD SYSTEM LOADING                                                     (e)
EMULATOR POD TEST                                                      (f)
**  RESET IN BY E7000 !                                                (g)
CLOCK = xx MHz
CPU MODE=H8S/xxxx OPERATION MODE=x(MDx-x=x) MODE SET=xxxxx             (h)
PIN MODE=xxx/xxx INTERNAL ROM SIZE=xxxxB  INTERNAL RAM SIZE=xxB
ADC MODE=xxxxxx  DMAC MODE=xxxxxx  REFRESH MODE=xxxxxx
SCI CHANNEL=xCH  16BIT TIMER MODE=TPUx-x
FAILED AT xxxx                                                         (i)
REMAINS EMULATION MEMORY      S = xxxxxx/E = xxxxxx                    (j)

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

(a) E7000 system program start message. Vn.m indicates the version number.

(b) IP address information file for the host computer 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 12-5 in section 12, Error Messages is displayed, the error may have occurred in the host computer connected via the FTP interface. Also check the host computer.

(c) Configuration file is being loaded. If an invalid configuration file is assigned, the following message is displayed:

```
***  54:INVALID CONFIGURATION FILE
```

If no configuration file is contained in the system disk, the following message is displayed:

```
***  55:CONFIGURATION FILE NOT FOUND
```

At this time, an unavailable system disk is inserted. Change system disks and re-initiate.

(d) 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

(e), (f) The emulator pod program is bein loaded and 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 incorrect 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, aborting operation

(iv) System programs for the emulator pod are not loaded because the incorrect E7000 system disk is inserted. 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 aborted (error code BK).

(g) The MCU is reset, the clock is set, and the specified clock type is displayed.

Note: (g) is not executed if an error has occurred in step (d), (e), or (f)

(h) The settings in the E7000 such as the selected MCU type and the MCU operating mode are displayed. To change the settings, use the MODE command. For details, refer to section 7.2.29, MODE.

(i) MCU pins are being checked. For details, refer to section 7.2.9, CHECK.

Note: (i) is not executed if an error has occurred in step (d), (e), or (f)

(j) The remaining emulation memory size that can be assigned.

(k) 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. 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) 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) ?
```

(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=xxxxxxxx

- 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

  No data will be displayed if an error does not occur.

(c) The E7000PC monitor enters command input wait state.

Notes: 1. Operation continues if an error occurs in step (b), but the error should be investigated according to section 5.3, Troubleshooting Procedure, without loading the E7000PC system program.

2. After message (c) is displayed, enter S or R to load the E7000PC system program that has been installed according to the procedure described in section 3.4.2, Installation in Part II, E7000PC Guide. If the system program is not correctly loaded, an error message will be displayed. In this case, check the IBM PC settings.

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   **

H8S/xxxx E7000 (HSxxxxEPDxxSF) Vn.m
Copyright (C) Hitachi, LTD. 199x                                          (a)
Licensed Material of Hitachi, Ltd.

CONFIGURATION FILE LOADING                                                (b)
HARD WARE REGISTER  READ/WRITE CHECK                                      (c)
POD SYSTEM LOADING                                                        (d)
EMULATOR POD TEST                                                         (e)
**  RESET IN BY E7000 !                                                   (f)
CLOCK = xx MHz
CPU MODE=H8S/xxxx OPERATION MODE=x(MDx-x=x) MODE SET=xxxxx
PIN MODE=xxx/xxx INTERNAL ROM SIZE=xxxxB  INTERNAL RAM SIZE=xxB           (g)
ADC MODE=xxxxxx  DMAC MODE=xxxxxx  REFRESH MODE=xxxxxx
SCI CHANNEL=xCH  16BIT TIMER MODE=TPUx-x
FAILED AT xxxx                                                            (h)
REMAINS EMULATION MEMORY      S= xxxxxx/E = xxxxxx                        (i)

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

(a) E7000PC system program start message. Vn.m indicates the version number.

(b) Configuration file is being loaded. If an invalid configuration file is assigned, the following message is displayed:

*** 54:INVALID CONFIGURATION FILE

If no configuration file is contained in the directory where the current system is stored, the following message is displayed:

*** 55:CONFIGURATION FILE NOT FOUND

This message indicates that the system software in the E7000PC is not correct. Re-install the correct system software and re-initiate the E7000PC.

(c) 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

(d), (e) The emulator pod program is being loaded and 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 incorrect 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 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, aborting operation

(iv) System programs for the emulator pod are not loaded because the wrong E7000PC system disk is inserted. 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 aborted (error code BK).

(f) The MCU is reset, the clock is set, and the specified clock type is displayed .

Note: (f) is not executed if an error has occurred in step (c), (d), or (e).

(g) The settings in the E7000 such as the selected MCU type and the MCU operating mode are displayed. To change the settings, use the MODE command. For details, refer to section 7.2.29, MODE.

(h) MCU pins are being checked. For details, refer to section 7.2.9, CHECK .

Note: (h) is not executed if an error has occurred in step (c), (d), or (e)

(i) The remaining emulation memory size that can be assigned.

(j) 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=xxxxxx
***  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 (HS2655TM01ME), and inform a Hitachi sales agency of the test results in detail because a system defect may be caused by a number of reasons.

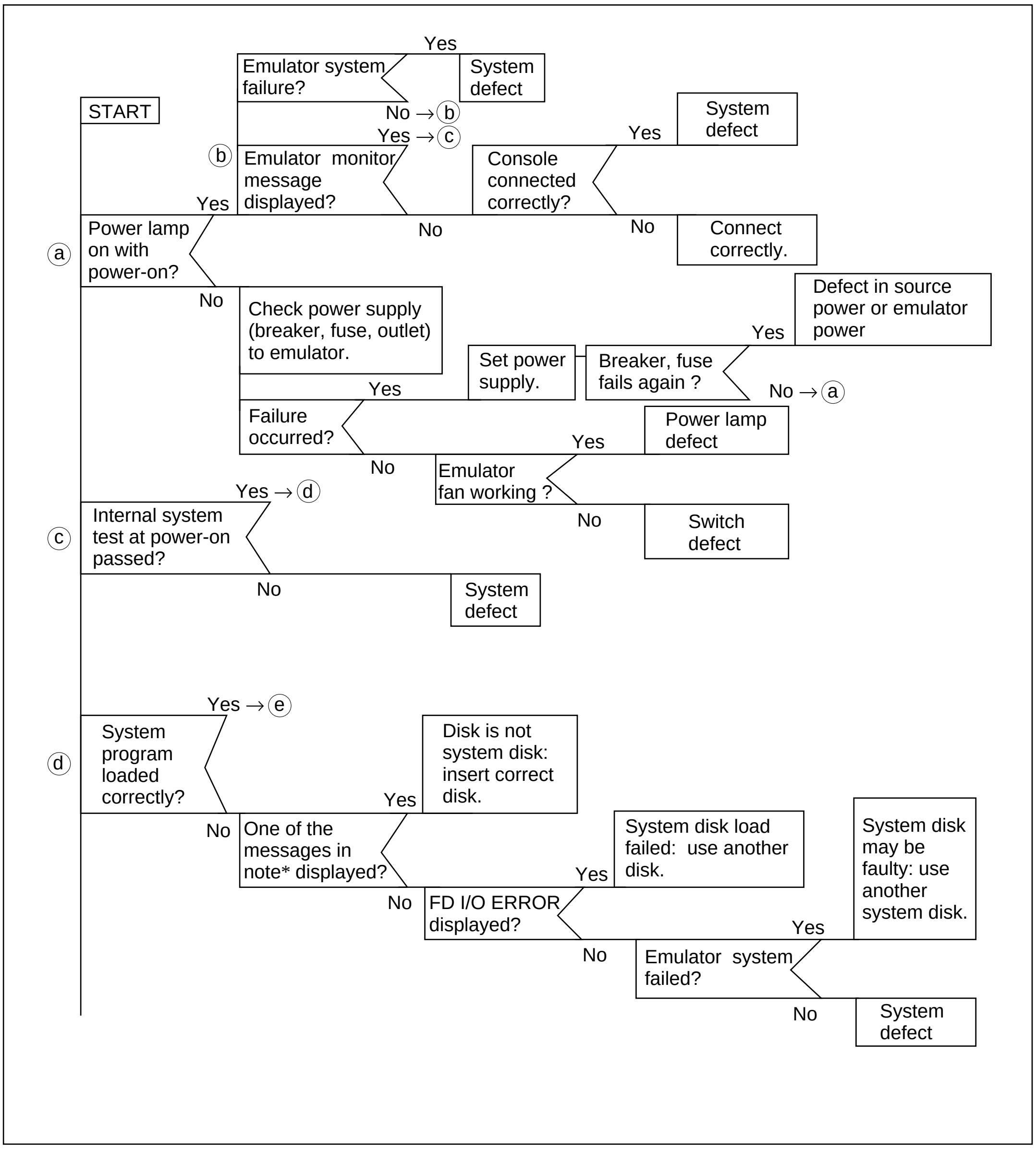
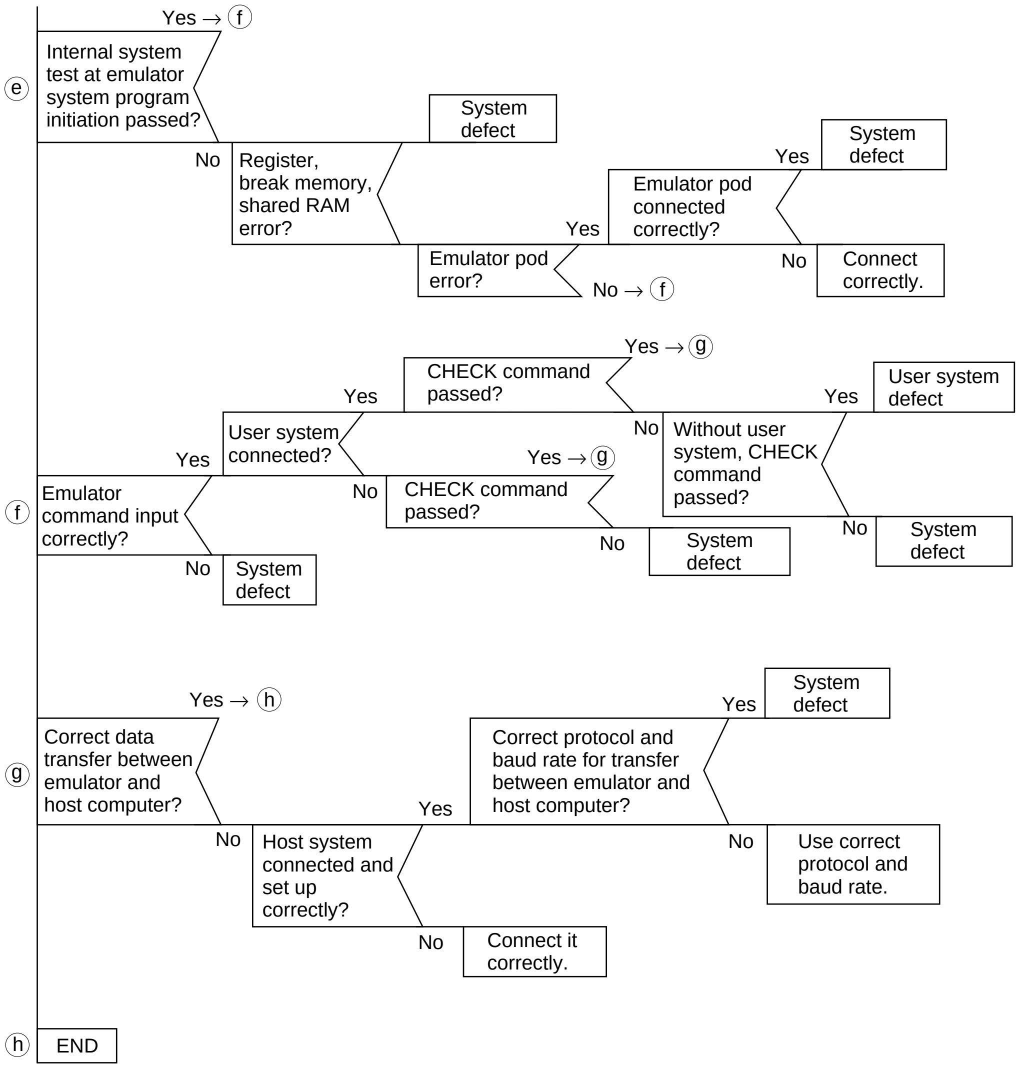

START
Yes
Emulator system failure?
System defect
No → b
Yes → c
b
Emulator monitor message displayed?
Console connected correctly?
Yes
System defect
Yes
No
No
Connect correctly.
a
Power lamp on with power-on?
No
Check power supply (breaker, fuse, outlet) to emulator.
Defect in source power or emulator power
Yes
Set power supply.
Breaker, fuse fails again ?
No → a
Yes
Failure occurred?
Yes
Power lamp defect
No
Emulator fan working ?
Yes → d
No
Switch defect
c
Internal system test at power-on passed?
No
System defect
Yes → e
d
System program loaded correctly?
Disk is not system disk: insert correct disk.
Yes
No
One of the messages in note* displayed?
System disk load failed: use another disk.
Yes
System disk may be faulty: use another system disk.
No
FD I/O ERROR displayed?
Yes
No
Emulator system failed?
No
System defect

Figure 5-1 Troubleshooting PAD



Yes → f
e
Internal system test at emulator system program initiation passed?
No
Register, break memory, shared RAM error?
System defect
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 computer?
No
Host system connected and set up correctly?
Yes
Correct protocol and baud rate for transfer between emulator and host computer?
Yes
System defect
No
Use correct protocol and baud rate.
No
Connect it correctly.
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

  Note: A hexadecimal constant is handled as signed 4-byte data. For example, H'FFFFFFFF is –D'1.

Symbols: There are two types of symbols; symbols defined with emulator's SYMBOL or ASSEMBLE command and that defined in cross software, such as an assembler or a C compiler. 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 uppercase characters can be defined by the ASSEMBLE command.

- Symbols defined in cross software

  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 the 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 1.12, Symbolic Debugging.

NOTE

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 emulator command (ASSEMBLE), 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 file is copied to another floppy disk. For details, refer to section 8.3, 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 termination	CTRL + C, (BREAK)	Terminates command execution. All characters typed so far are lost and the emulator enters command input wait state.

6.2.2 Display Control

• Display stop	CTRL + S	Temporarily stops display. Resumes display by entering CTRL and Q keys.
• Display restart	CTRL + Q	Resumes 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 resumes 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 another command :D.(RET) :D 1000 1010 Displays the parameters specified in the previous DUMP command execution and enters command input wait state.

6.2.4 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, trace, performance analysis, and coverage. Table 7-1 lists the emulation commands that enable these functions.

Table 7-1 Emulation Commands

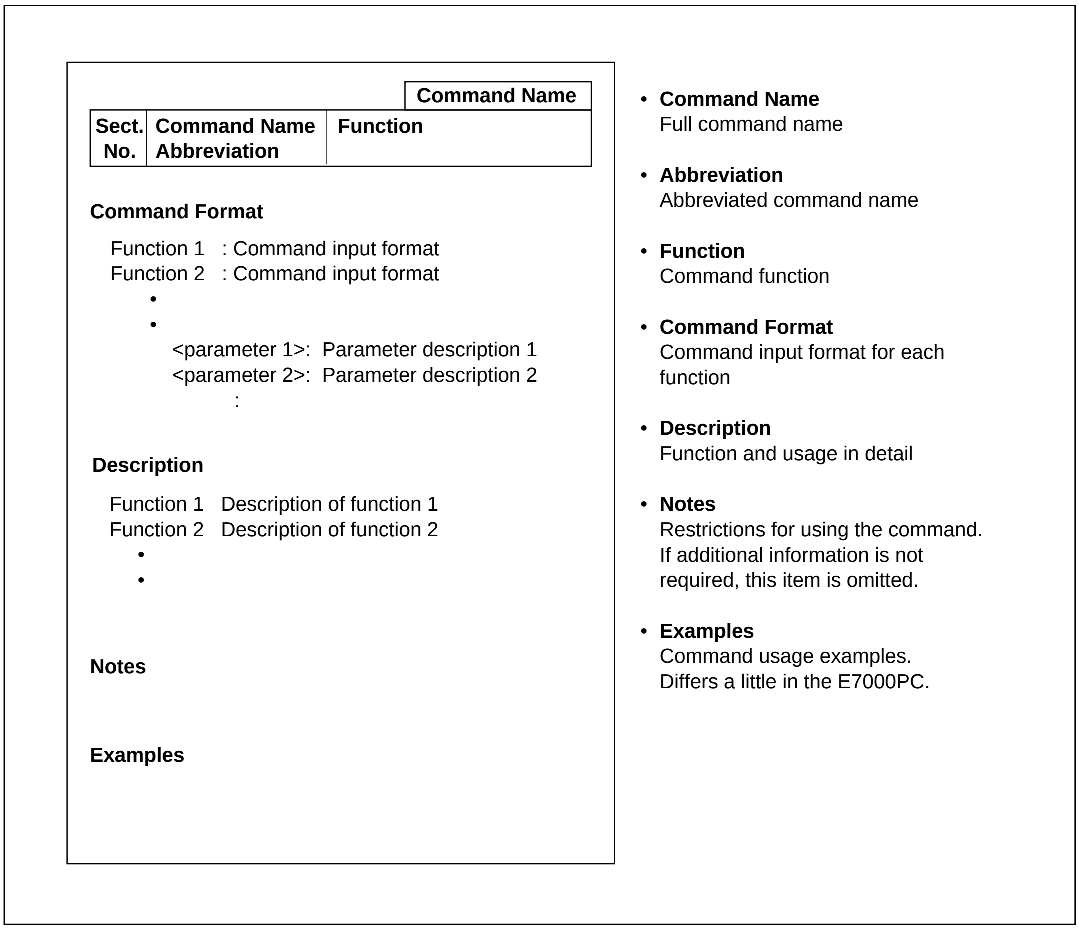
Command	Function	Usable/Unusable in Parallel Mode
.<register>	Modifies and displays register contents	Unusable
!<symbol> or &<symbol>	Displays symbol value	Usable
ABORT	Terminates emulation in parallel mode	Usable
ASSEMBLE	Assembles program one line each	Unusable
BACKGROUND_ INTERRUPT	Sets and displays user interrupt accept status in command wait state	Unusable
BREAK	Sets, displays, and cancels software breakpoints	Only display function is available
BREAK_CONDITION 1,2,3,4,5,6	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 MCU 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
DISPLAY_COVERAGE	Displays coverage trace results	Unusable
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 input command history	Usable
ID	Displays the emulator program version number	Usable
LED1,2,3,4	Specifies, displays, and cancels memory content display on LEDs	Usable

Table 7-1 Emulation Commands (cont)

Command	Function	Usable/Unusable in Parallel Mode
LED_OUT1,2	Specifies and displays analog output of LED display data	Usable
MAP	Specifies and displays memory attribute	Unusable
MEMORY	Modifies and displays memory contents	Usable
MODE	Specifies and displays the MCU type and MCU operating mode	Unusable
MOVE	Transfers memory contents	Unusable
MOVE_TO_RAM	Moves ROM contents to standard emulation memory	Unusable
PERFORMANCE_ ANALYSIS	Specifies, cancels, initializes, and displays performance measurement data	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 MCU	Unusable
RESULT	Displays execution results	Unusable
SET_COVERAGE	Initializes the coverage trace function	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 acquisition mode	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**
  Restrictions 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 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.

.<register>

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, CCR, EXR, MACH, MACL
(General-purpose register) ER0, ER1, ER2, ER3, ER4, ER5, ER6, ER7 (SP)
R0, R1, R2, R3, R4, R5, R6, R7
E0, E1, E2, E3, E4, E5, E6, E7
R0H, R0L, R1H, R1L, R2H, R2L, R3H, R3L,
R4H, R4L, R5H, R5L, R6H, R6L, R7H, R7L

<data>: The value to be set in the specified register

Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and they cannot be specified with this command.

Description

- Modification

— Direct mode

Sets the specified value in the specified register. SP can be specified instead of ER7.

```
: .<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):

ER0, ER1, ER2, ER3, ER4, ER5, ER6, ER7 (SP), PC, CCR, EXR, MACH, MACL

Note: In interactive mode, the only general-purpose register that can be specified is ERn.

Example: To modify registers in interactive mode

```
: .<register> (RET)
 <register>        =xxxxxxxx ?      yyyy (RET)
 <register>        =xxxxxxxx ?      yyyy (RET)
      .                  .               .
      .                  .               .
```

yyyy <data>: Inputs the value to be newly 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:

ER0 to ER6:	H'00000000
ER7 (SP):	Internal I/O start address – 2
PC:	Reset vector value
CCR:	H'80
EXR:	H'07
MACH:	00000000
MACL:	00000000

If the MCU is reset by the emulator RESET or CLOCK command, registers are set as follows:

ER0 to ER6:	Value before reset
ER7 (SP):	Value before reset
PC:	Reset vector value
CCR:	H'80
EXR:	H'07
MACH:	Value before reset
MACL:	Value before reset

Since the reset values for ER0 to ER7, MACH, and MACL in the MCU are not fixed, the initial values must be set by a program.

.<register>

Examples

1. To set H'5C60 in the PC, H'FFE00 in the SP, H'FF in R1, and H'11 in R2H, and H'1 in E0, and then display all registers:

```
:.PC 5C60  (RET)
:.SP FFE00  (RET)
:.R1 FF  (RET)
:.R2H 11  (RET)
:.E0 1  (RET)
:R  (RET)
 PC=00005C60 CCR=80:I******* EXR=07:*****210
 MACH=00000000 MACL=00000000
 ER0-ER3  00010000 000000FF 00001100 00000000
 ER4-ER7  00004000 00005000 00000000 000FFE00
:
```

2. To modify the contents of control registers in interactive mode:

```
:.PC (RET)
 PC     =00001000 ? 2000 (RET)
 CCR    =80:I******* ? 00 (RET)
 EXR    =07:*****210 ? . (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 the attribute type.

Table 7-2 Symbol Value Display Formats

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 symbol if that line number symbol is specified.

!<symbol>

Examples

1. To disassemble the contents of an address specified by a symbol:

```
:DISASSEMBLE !MAIN_START (RET)
   ADDR      CODE  LABEL    MNEMONIC  OPERAND
 00000500 1C89     !MAIN_START
                            CMP.B     #8,R1L
 00000502 4706              BEQ       !MAIN_001
 00000504 6AA800FF          MOV.B     R0L,@00FFFF00:32
          FF00
 00000506 6B02FD80          MOV.W     @FD80:16,R2
    :        :                :           :
```

2. To display the contents of array msym/sym_chr:

```
:!msym/sym_chr (RET)
  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 GO command execution in parallel mode (prompt #), and cancels parallel mode.

— When GO command execution is terminated by the ABORT command in parallel mode, BREAK KEY is displayed as the termination cause.

Note

Similar to when pressing the break key, when executing the ABORT command in a command chain file, the emulator outputs the termination cause followed by the confirmation message STOP COMMAND CHAIN (Y/N).

Example

To terminate GO command emulation in parallel mode:

```
:GO RESET (RET)
 ** PC=00001022    (RET)           (To enter parallel mode)
#ABORT (RET)
 PC=00001300 CCR=80:I******* EXR=07:*****210
 MACH=00000000 MACL=00000000
 ER0 - ER3 00000000 00001000 00002000 00003000
 ER4 - ER7 00044000 00225000 00006000 000FFF7E
 RUN-TIME=D'0000H:00M:01S:457355US
 +++:BREAK KEY
:
```

7.2.4	**ASSEMBLE** **A**	**Assembles program one line each**

Command Format

- Line assembly : ASSEMBLE Δ<address> (RET)

  <address>: The address where the object program is to 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 which are written to memory. Assembly is continued until a period (finishing subcommand) is entered. The input and output formats are as follows:

```
: ASSEMBLE  <address>   (RET)
   ADDR        CODE      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 the displayed address.
/[<address 1>[Δ<address 2>]]	Disassembles instructions from <address 1> to <address 2> and displays them. If <address 2> is omitted, the first 16 instructions from <address 1> are displayed. If only a slash (/) is input, the contents from the ASSEMBLE command start address to the current address –1 are disassembled.
(RET) only	Increments the address and enters subcommand input wait state.
^	Decrements the address (Odd address –1, Even address –2), and enters subcommand input wait state.
:	Increments 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 referred during line assembly,

```
*** 33:INVALID ASM OPERAND
```

is displayed.

— Even if an odd address is specified, machine codes are written to memory. In that case, the following warning message is displayed:

```
*** 82:ODD ADDRESS
```

— When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, but assembly statement input and disassembly display are performed.

ASSEMBLE

Example

To perform line assembly from address H'1000:

```
:A 1000 (RET)
   ADDR         CODE     LABEL     MNEMONIC  OPERAND
 00001000      0000               NOP
 00001000      0000  ? SUB010    MOV.W     #1,R3 (RET)
 00001004      0000  ?           ADD.B     R0L,R1L (RET)
 00001006      0000  ?           MOV.W     R0,@FB80 (RET)
 0000100C      0000  ? .(RET)
:
```

7.2.5	BACKGROUND_INTERRUPT BI	Sets and displays user interrupts in command wait state

Command Format

- Setting : BACKGROUND_INTERRUPTΔ[[<mode>][:<loop program address>]] [;C] (RET)
- Display : BACKGROUND_INTERRUPT (RET)

<mode>: User interrupt accepting mode in command wait state
E: Enables user interrupts in command wait state
D: Disables user interrupts in command wait state
(Default at shipment)

<loop program address>: Address of the loop program for accepting user interrupts
(at shipment, the last address of internal RAM area - 1)

C: Writes the setting contents to the configuration file

Description

- Setting

— Enables user interrupts in command wait state and sets the address of the loop program for accepting user interrupts. The loop program must be in the RAM area. If no address is specified, the address specified before is used. After setting, the loop program execution starts.

```
: BACKGROUND_INTERRUPT:FFEC00   (RET)
```

— Disables user interrupts in command wait state.

```
: BACKGROUND_INTERRUPT  D   (RET)
```

— The following message is output when the C option is specified:

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

When using the E7000 and Y is input, the specifications are written in a configuration file on the system disk. When using the E7000PC and Y is input, specifications are written in the configuration file in the directory where the current system program is stored on the IBM PC. Hereafter, when the E7000 is activated with the system disk, the saved specifications go into effect.

BACKGROUND_INTERRUPT

- Display

Displays user interrupt accepting mode and the executing address of the loop program for accepting user interrupts. If a break has occurred during user interrupt processing and the loop program has been stopped, the register values at termination and the cause of termination are displayed.

```
: BACKGROUND_INTERRUPT   (RET)
 USER_INTERRUPT = x  LOOP_PROGRAM_ADDRESS = yyyyyyyy
 [<termination information>]
```

x: User interrupt accepting mode

E: User interrupts are enabled (the loop program is being executed)

D: User interrupts are disabled (the loop program has been stopped)

S: A break has occurred during user interrupt processing (the loop program has been stopped)

yyyyyyyy: Address of loop program for accepting user interrupts

<termination information>: Register values at loop program termination and termination cause (displayed only when S is displayed above)

Display format:

```
-PC=00000000 CCR=80:I******* EXR=07:*****210
-MACH=00000000 MACL=00000000
-ER0-ER3  00000000 00000001 00000002 00000003
-ER4-ER7  00000004 00000005 00000006 00000007
 +++:<cause of termination>
```

Table 7-4 Termination Causes of Loop Program for Accepting User Interrupts

Display	Cause of Termination
GUARDED AREA ACCESSED	A guarded memory area was accessed
WRITE PROTECT	A write-protected area or internal ROM area was written to
ILLEGAL INSTRUCTION	A break instruction (H'5740 to H'577F) was executed
RESET IN BY E7000	The program was terminated with the RES signal because an error has occurred in the user system
LOOP PROGRAM ADDRESS IS NOT IN RAM	The executing area of the loop program for accepting interrupts is not in the RAM, therefore, the loop program cannot be executed

Notes

1. In command wait state, BRA $ instruction (instruction code: H'40FE) is set to the address of the loop program for accepting user interrupts and executed, therefore, note the following.

   (a) Do not specify the address of the loop program for accepting user interrupts in the ROM area. If the specified address is in the ROM area, the loop program cannot be executed. Specify an address within the RAM area and enable user interrupts again.

   (b) Specify the address of the loop program for accepting user interrupts within the area that is not used by user program. The loop program uses a 2-Mbyte area.

   (c) When the address of the loop program for accepting user interrupts is specified, the memory contents before this specification are not stored. Therefore, the contents of the loop program address is an BRA $ instruction even after user interrupts are disabled or after the loop program address is changed.

2. When one of the causes of termination listed in table 7-4 occurs during interrupt processing in command wait state, the interrupt processing stops there. If an emulation command is executed in this state, the following message is displayed after the emulation command execution. In this case, either change the interrupt processing program and enable user interrupts, or disable user interrupts.

```
*** 69: STOPPED THE BACKGROUND INTERRUPT
```

3. When a GO, STEP, or STEP_OVER command is entered during user interrupt program execution, the emulator forcibly terminates the program and the GO, STEP, or STEP_OVER command is executed. After the program has stopped, the loop program for accepting user interrupts is initiated again. The loop program uses the register values when the GO, STEP, or STEP_OVER command execution was terminated. In the same way, when a register value is changed by the .<register> command or the RESET command is executed during user interrupt program execution, the emulator forcibly terminates the program and then initiates the loop program for accepting user interrupts again.

4. Do not use this command when using the system, such as an OS, that does not return the processing from the user interrupt routine to the routine where the interrupt has occurred. If used, execution does not return to the loop program for accepting user interrupts after the user interrupt processing.

5. Do not generate a reset exception when user interrupts are enabled. If generated, the user program is initiated and execution does not return to the loop program for accepting user interrupts.

6. During user interrupt program execution in command wait state, the software breakpoints (BREAK and BREAK_SEQUENCE) and hardware break settings (BREAK_CONDITION1,2, 3,4,5,6) become invalid, and no trace information is acquired.

Examples

1. To specify the executing address of the loop program for accepting user interrupts to H'EC00, and begin to accept user interrupts in command wait state:

```
:BI E:EC00  (RET)
:
```

2. To display the current user interrupt accepting mode in command wait state:

```
:BI  (RET)
 USER_INTERRUPT=S LOOP_PROGRAM_ADDRESS=0000EC00
-PC=00005C60  CCR=80:I*******  EXR=07:*****210
-MACH=00000000  MACL=00000000
-ER0 - ER3  00000000 00001000 00002000 00003000
-ER4 - ER7  00044000 00225000 00006000 000FFF7E
 +++:WRITE PROTECT
:
```

7.2.6	**BREAK** **B**	**Sets, displays, and cancels software breakpoints**

Command Format

- Setting : BREAKΔ<breakpoint to be set>[,<breakpoint to be set>]... (RET)
- Display : BREAK (RET)
- Cancellation : BREAK[Δ]–[<breakpoint to be cancelled>
  [,<breakpoint to be cancelled>]...] (RET)

<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>: 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'5770). 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 emulation memory) because the specified address contents is replaced with a break instruction. Do not set software breakpoints at any of the addresses below:

- Addresses specified with the BREAK_SEQUENCE command
- Addresses that hold break instructions (H'5770)
- Addresses in internal I/O area

— The user can also specify the number of times a breakpoint must be reached.

Example: To break when the instruction at address H'300 has been executed five times.

```
:BREAK 300 5 (RET)
```

Note: When multiple passes are specified for a breakpoint, the program temporarily stops each time a software breakpoint is passed to update the pass count. 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 the instruction replaced with a brek instruction cannot be restored when user program execution is terminated, the breakpoint setting is cancelled but the break instruction remains as an illegal instruction in the user program.

- 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 until GO command termination, in hexadecimal)
(d) Symbol (only displayed if the software breakpoint address has a symbol)

Note: The pass counter is cleared by the next GO command.

- 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–<breakpoint>[,<breakpoint> ]...(RET)
```

- — Cancellation of all breakpoints.

```
: BREAK– (RET)
```

Notes

1. A software breakpoint must be set at the start address of the MCU instruction. If not, a break does not occur and instructions will not be executed correctly.

2. 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.

3. If the contents of the address where a breakpoint is set is modified by user program execution, the setting is cancelled after user program termination.

4. If the memory contents are modified by the LOAD command, breakpoints are not cancelled.

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
 000030F0        0001       0000     !symbol
:
```

BREAK

4. To cancel a software breakpoint:

```
:B — 1000 (RET)
:
```

5. To cancel all software breakpoints:

```
:B — (RET)
:
```

7.2.7	BREAK_CONDITION1,2,3,4,5,6 BC1,2,3,4,5,6	Specifies, displays, and cancels a hardware break condition

Command Format

- Setting : BREAK_CONDITION (1/2/3/4/5/6)Δ<condition>[[Δ <condition >]
  [Δ<condition>]...] (RET)
- Display : BREAK_CONDITION [(1/2/3/4/5/6)] (RET)
- Cancellation : BREAK_CONDITION [(1/2/3/4/5/6)] [Δ]– (RET)

(1/2/3/4/5/6): Break command number
When omitted, 1, 2, 3, 4, 5, and 6 will all be displayed or cancelled.
<condition>: Hardware break condition (refer to tables 7-4 to 7-8 for details)

Description

- Setting

— Specifies hardware break conditions. Program execution stops when the specified conditions are satisfied. Combinations of BREAK_CONDITION1,2,3,4 can be specified with the GO command option. In such combinations, normal mode, sequential break mode 1, 2, 3, or time measurement mode 1, 2 can be selected as the break mode. For details on specification and mode operation, refer to section 7.2.21, GO.

The relationship between the GO command option and BREAK_CONDITION1,2,3,4,5,6 command is summarized in table 7-5.

Table 7-5 Relationship between GO Command Option and BREAK_CONDITION1,2,3,4,5,6 Command

GO Command Option	BC1	BC2	BC3	BC4	BC5	BC6
Normal mode (no option specified)	O	O	O	O	O	O
Sequential break mode 1 (S1)	❑	❑	X	X	X	X
Sequential break mode 2 (S2)	❑	❑	❑	X	X	X
Sequential break mode 3 (S3)	❑	❑	❑	❑	X	X
Time measurement mode 1 (I1)	❑	❑	X	X	X	X
Time measurement mode 2 (I2)	❑	❑	X	X	X	X

Symbols: ❑ Must be specified.
O Specification is valid.
X Specification is invalid.

BREAK_CONDITION1,2,3,4,5,6

— Conditions specified by BREAK_CONDITION1,2,3,4,5,6 are listed in tables 7-6 to 7-8. Note that the specifiable conditions for BREAK_CONDITION1, BREAK_CONDITION2,3,4, and BREAK_CONDITION5,6 are different.

Table 7-6 Specifiable Conditions (BREAK_CONDITION1)

Item and Input Format	Description
Address condition A=<address 1>[:<address 2>][;NOT]	The condition is satisfied when the address bus value is in the range from <address 1> to <address 2>. 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.*

Data condition
D = <1-byte value>[;NOT]
WD = <2-byte value>[;NOT]
LD = <1-byte value> [;NOT]
(Values on data bus from D7 to D0)
HD = <1-byte value>[;NOT]
(Values on data bus from D15 to D8)
	The condition is satisfied when the data bus value matches the specified value. D, LD, and HD are valid when byte access is performed, while WD is valid when word access is performed. If NOT is specified, the condition is satisfied when data other than the specified one is accessed.*

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 PRG: Program fetch cycle DAT: Execution cycle DMA: DMA cycle DTC: DTC cycle Default: All the above bus cycles	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.
External probe condition PROB=<value> <value>: Values for probes 1 to 8	The condition is satisfied when all the emulator's external probe signals match the specified values. Specify <value> as one byte of data. Each bit corresponds to a probe number, as follows: 7 6 5 4 3 2 1 0 ← Bit position x x x x x x x x ← Specified value | | | | | | | | 8 7 6 5 4 3 2 1 ← Probe number x: 0=Low level 1=High level (Ex.) To generate a break when probes 1 and 6 are high and the others are low, specify: PROB=H'21 External probe 8 is used for a multibreak. Specify EXECUTION_MODE command PB8 option and specify break when probe 8 is low with this condition. For details on multibreak, refer to the description on external probe values in section 5.4.1.

Note: Refer to the address and data condition descriptions on the following pages.

Table 7-6 Specifiable Conditions (BREAK_CONDITION1) (cont)

Item and Input Format	Description
External interrupt condition NMI [:L] or NMI:H IRQ=<value>	• The condition is satisfied when NMI is at the specified level. — NMI:L or NMI: Condition is satisfied when NMI is low. — NMI:H: Condition is satisfied when NMI is high. • The condition is satisfied when all IRQn pins are at the specified levels. Specify <value> as one byte of data. Each bit corresponds to a signal as follows. 7 6 5 4 3 2 1 0 ← Bit position x x x x x x x x ← Specified value | | | | | | | | 7 6 5 4 3 2 1 0 ← IRQ number x: 0=Low level 1=High level (Ex.) To generate a break when IRQ1 and IRQ5 are high and the remaining IRQ pins are low, specify: IRQ=H'22 • Both NMI and IRQ conditions can be specified simultaneously.
Number of times a condition is satisfied COUNT=<value> <value>: H'1 to H'1000	This condition can be specified in combination with any of the above conditions. The complete condition combination is satisfied when the specified condition has been satisfied for the specified number of times.
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 have been executed after the other specified condition is satisfied. When this conditions is specified in combination with the number-of-times conditions, the complete condition combination is satisfied when the specified number of bus cycles have been executed after the specified condition has been satisfied for the specified number of times.

BREAK_CONDITION1,2,3,4,5,6

Table 7-7 Specifiable Conditions (BREAK_CONDITION2,3,4)

Item and Input Format	Description
Address condition A=<address>	The condition is satisfied when the address bus value matches the specified value.*

Data condition
D = <1-byte value>
WD = <2-byte value>
LD = <1-byte value>
(Values on data bus from D7 to D0)
HD = <1-byte value>
(Values on data bus from D15 to D8)
	The condition is satisfied when the data bus value matches the specified value. D, LD, and HD are valid when byte access is performed, while WD is valid when word access is performed.*

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 PRG: Program fetch cycle DAT: Execution cycle DMA: DMA cycle DTC: DTC cycle Default: All bus cycles described above	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.
External probe condition PROB=<value>	The condition is satisfied when all the emulator's external probe signals match the specified values. Specify this condition in the same way as BREAK_CONDITION1 shown in table 7-6.
External interrupt condition NMI [:L] or NMI:H IRQ=<value>	The condition is satisfied when NMI or IRQ is at the specified level. Specify this condition in the same way as BREAK_CONDITION1 shown in table 7-6.

Note: Refer to the address and data condition descriptions on the following pages.

Table 7-8 Specifiable Conditions (BREAK_CONDITION5,6)

Item and Input Format	Description
PC conditions PC=<address>	The condition is satisfied when the instruction at the specified address is executed.
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).

— The address conditions and data conditions are satisfied when the address bus and data bus values match the specified values. Note the following when specifying break conditions.

- Word access

  Word data is accessed in one bus cycle. Only WD (word data) is valid for the data condition.

- Byte access

  All addresses can be accessed byte access. Both even and odd address values are valid for the address condition while only D (byte data) is valid for the data condition.

Notes: 1. In the actual MCU, data bus lines D15 to D8 are used when an 8-bit access area is accessed. However, in the emulator, D15 to D8 are used when an even address is accessed, and D7 to D0 are used when an odd address is accessed. When specifying data in byte access cycles, follow note 2 below regardless of whether a 16-bit bus area or an 8-bit bus area is accessed.

2. D, HD, or LD can be specified as byte data. Use these three data types depending on the trace conditions to be specified.

D: If the specified address is even, data on data bus D15–D8 is specified. If the specified address is odd, data on data bus D7–D0 is specified. If no address is specified or if an address range or mask is specified, data on data bus D15–D8 is automatically specified as shown below.

Example 1: BC1 A=101 D=10

A condition is satisfied when byte data H'10 is written to or read from address H'101.

Example 2: BC1 A=100:1FF D=20

A condition is satisfied when byte data H'20 is written to or read from odd addresses from H'101 to H'1FF.

HD: Byte data (data bus D15–D8) access to an even address is always specified. Data access to an odd address is ignored.

Example: BC1 A=1000:10FF HD=80

A condition is satisfied if byte data H'80 is written to or read from even addresses from H'1000 to H'10FF.

LD: Byte data (data bus D7–D0) access to an odd address is always specified. Data access to an even address is ignored.

Example: BC1 A=1000:10FF LD=80

A condition is satisfied if byte data H'80 is written to or read from odd addresses from H'1000 to H'10FF.

Note that D, HD, or LD cannot be specified in word access.

3. When an 8-bit access area is accessed in word units, two byte access cycles are executed in the actual MCU. However, in the emulator, it is executed in one bus cycle. In this case, byte data (D, HD, and LD) conditions are not satisfied. Specify word data (WD) conditions.

— Bit masks can be specified for data, IRQ, or external probe conditions in 1-bit, or 4-bit units. When a bit is masked, the condition is satisfied irrespective of its bit value. To mask a bit, specify it as * at input. The number of bits that can be masked differs depending on the conditions. Examples of masks are shown below and in table 7-9.

Example 1: A condition is satisfied when the D0 bit is 0 in a byte data condition.

```
: BREAK_CONDITION 1   D = B'******* 0 (RET)
```

Example 2: A condition is satisfied when IRQ2 is 0 in the IRQ condition.

```
: BREAK_CONDITION 2   IRQ = B'*****0** (RET)
```

Table 7-9 Mask Specifications (BREAK_CONDITION1,2,3,4)

Radix	Mask Unit	Example	Mask Position	Allowed Condition
Binary	1 bit	B'01*1010*	D0 and D5 bits	BREAK_CONDITION1 data (WD, D, HD, LD), BREAK_CONDITION1,2,3,4 IRQ, PROB, BREAK_CONDITION5,6 address
Hexadecimal	4 bits	H'F*50	D15 to D12 bits	BREAK_CONDITION1,2,3,4 data (WD, D, HD, LD), IRQ, PROB

Notes 1. BREAK_CONDITION1 data and BREAK_CONDITION5,6 address are masked in bit units and BREAK_CONDITION2,3,4 data is masked in 4-bit units.

2. BREAK_CONDITION1 address is masked from the low-order bit. However, the optional bit position or the range specification can not be masked. In addition, BREAK_CONDITION2,3,4 address cannot be masked.

- **When the specification is possible**
  BREAK_CONDITION 1 A=H'10**

- **When the specification is impossible**
  BREAK_CONDITION 1 A=H'1*00
  BREAK_CONDITION 1 A=H'100*:10**

— If a hardware break condition is specified and that condition is satisfied, emulation stops after at least one instruction has been executed.

BREAK_CONDITION1,2,3,4,5,6

- Display

Displays specified conditions. The specified input character string is displayed as is before. All six break conditions are displayed if break command numbers 1 to 6 are omitted. For BREAK_CONDITION1, delay count since the previous break condition was satisfied is displayed. If no BREAK_CONDITION1 condition is specified, the delay count is not displayed.

```
: BC (RET)
 BC1 (BREAK_CONDITION1 command setting)
 PASS COUNT = xxxx DELAY COUNT = yyyy      xxxx:  Pass count
 BC2 (BREAK_CONDITION2 command setting)    yyyy:  Delay count after the condition is satisfied
 BC3 (BREAK_CONDITION3 command setting)
 BC4 (BREAK_CONDITION4 command setting)
 BC5 (BREAK_CONDITION5 command setting)
 BC6 (BREAK_CONDITION6 command setting)
```

Note: Pass count stops when the condition is satisfied. Therefore, the pass count whose condition is satisfied is not counted from the conditional satisfaction to the program stop.

- Cancellation

Cancels specified conditions. When conditions 1, 2, 3, 4, 5, and 6 are omitted, all break conditions are cancelled.

— Cancels all break conditions.

```
: BREAK_CONDITION – (RET)
```

— Cancels BREAK_CONDITION2 conditions.

```
: BREAK_CONDITION2 – (RET)
```

Examples

1. To generate a break when byte data H'10 is accessed at address H'FF00:

```
:BC1 A=FF00 D=10  (RET)
:
```

2. To generate a break when data is written from address H'F000 to address H'FFFF:

```
:BC1 A=F000:FFFF  (RET)
:
```

4. To generate a break when IRQ0 signal is low:

```
:BC2 IRQ=B'*******0  (RET)
:
```

4. To display the specified condition:

```
:BC  (RET)
 BC1 A=FF00 D=10
 PASS COUNT=0000 DELAY COUNT=0000
 BC2 IRQ=B'*******0
 BC3
 BC4
 BC5
 BC6
:
```

5. To delete the specified condition:

```
:BC1 —  (RET)
:BC2 —  (RET)
:
```

7.2.8	**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: When the specified address is odd, it is rounded down to an even address.

Description

- Setting

— Sets pass points to enable the break for which the pass sequence is specified (sequential break). GO command emulation terminates when these pass points have been passed in the specified sequence.

— 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 whose contents is value H'5770
- Addresses in the internal I/O area

— Pass points or a reset point are ignored during STEP and STEP_OVER command execution. Therefore, the pass count is not updated during STEP and STEP_OVER command execution.

— If the instruction replaced with a break instruction cannot be restored when user program execution is terminated, the breakpoint is cancelled but the break instruction remains as an illegal instruction in the user program.

- 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)
```

Notes

1. The pass points or reset point must be set at the start address of an MCU instruction. If not, a break does not occur and instructions are not executed correctly.

2. In parallel mode, if a command (for example, memory access) is executed and the emulation stops at a pass point at the same time, command execution 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.

BREAK_SEQUENCE

3. If the contents of the address where a breakpoint is set is modified by user program execution, the setting is cancelled after user program termination.

Examples

1. To set pass points at addresses H'4000, H'4100, H'4200, and H'4300 in that order and a reset point at address H'2000:

```
:BS 4000 4100 4200 4300 (RET)
:BS 2000 ;R (RET)
:
```

2. To display the specified pass points and reset point:

```
:BS  (RET)
 PASS POINT NO1 = 00004000  0000
 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)
:
```

7.2.9	**CHECK** **CH**	**Tests MCU pins**

Command Format

- Test : CHECK (RET)

Description

- Test

Tests the status of the MCU pins shown in table 7-10.

Table 7-10 MCU Pin Test

Pin Name	Error Status	Remarks
RES	RES signal is fixed low	
NMI	NMI signal is fixed low	
STBY	STBY signal is fixed low	
WAIT	WAIT signal is fixed low	Not tested in single-chip mode
BREQ	BREQ signal is fixed low	Not tested in single-chip mode
IRQ0	IRQ0 signal is fixed low	
IRQ1	IRQ1 signal is fixed low	
IRQ2	IRQ2 signal is fixed low	
IRQ3	IRQ3 signal is fixed low	
IRQ4	IRQ4 signal is fixed low	
IRQ5	IRQ5 signal is fixed low	
IRQ6	IRQ6 signal is fixed low	
IRQ7	IRQ7 signal is fixed low	

If an error occurs,

```
FAILED AT <pin name>
```

is displayed.

Note

Some signal lines in table 7-10 are multiplexed with I/O ports, and may have different pin functions according to the MCU control register values. The emulator always tests these pins as control signal lines without checking the control register values.

CHECK

Example

When the IRQ0 signal is low:

```
    :CH  (RET)
     FAILED AT IRQ0
    :
```

7.2.10	CLOCK CL	Sets or displays clock

Command Format

- Setting : CLOCKΔ<clock> (RET)
- Display : CLOCK (RET)

<clock>: One of the following signals:

8:	8-MHz emulator internal clock
20:	20-MHz emulator internal clock
U:	User system clock signal
X:	Emulator pod crystal oscillator CLOCK signal

Description

- Setting

  — Selects emulator clock signals from the user system or from the emulator clock (installed in the emulator). Resets the MCU (with an RES signal input) when a clock is selected, and consequently, internal I/O registers and control registers return to their reset values. However, by controlling the EAE bit of the bus control register L (BCRL), the emulator enables access to all internal ROM areas.

  — Displays the specified clock signal. If the user system clock (U) or the crystal oscillator clock (X) is specified, but the clock signal is not input, an error occurs and the 8-MHz emulator clock (8) is set instead. At emulator initiation, the user system, crystal oscillator, and 8-MHz emulator clocks are selected in that order, and the correct clock signal is set.

- Display

  Displays the current clock signal.

```
: CLOCK  (RET)
  CLOCK = <Used clock>
```

<Used clock>:

8:	8-MHz emulator internal clock
20:	20-MHz emulator internal clock
USER:	User system clock
X'TAL:	Emulator pod crystal oscillator

CLOCK

CAUTION

1. **If U or X is specified and the following clock signal problems occur, the emulator system program will terminate.**
   In this case, the emulator system program must be restarted.
   - **User system clock is not input when it is specified.**
     (Vcc does not have problem.)
   - **Crystal oscillator clock is not input when it is specified.**
2. **When the user system interface cable is connected to the emulator, power must be supplied from the user system. If it is not, this command cannot be executed regardless of the current clock type. In addition, note that the user system interface cable must not be connected or disconnected at emulator initiation.**

Examples

1. To use the user system clock signal:

```
:CL U (RET)
 ** RESET IN BY E7000 !
 CLOCK = USER
:
```

2. To use the 20-MHz emulator clock signal:

```
:CL 20 (RET)
 ** RESET IN BY E7000 !
 CLOCK = 20 MHz
:
```

3. To display the current clock signal:

```
:CL (RET)
 CLOCK = 20 MHz
:
```

7.2.11	COMMAND_CHAIN CC	Inputs emulator command from a file (specific to the E7000)

Command Format

- Command input : COMMAND_CHAINΔ<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    0   1FFFF;U
                 MEMORY   100
                 30
                 .
                 CLOCK
```

```
Execution results:  :  COMMAND_CHAIN   <file name>  (RET)
                    :  MAP    0    1FFFF;U
                       REMAINS EMULATION MEMORY     S=60000/E=000000
                    :  MEMORY      100
                       00000100       00                  ?   30
                       00000101       00                  ?   .
                    :  CLOCK
                       CLOCK  =  USER
                    :  (Command input wait state)
```

COMMAND_CHAIN

— 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 computer 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, Emulation Support Function, in Part II, E7000PC 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.

7.2.12	**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 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. For details, refer to section 7.2.31, RADIX.

CONVERT

Examples

1. To convert hexadecimal data (H'7F):

```
:CV H'7F  (RET)
 H'7F D'127 Q'177 B'1111111   ....
:
```

2. To convert the expression:

```
:CV H'31+D'16  (RET)
 H'41  D'65 Q'101 B'1000001   ...A
:
```

7.2.13	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; continues to search for another occurrence of the specified data. 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.

DATA_CHANGE

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)
:
```

7.2.14	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: H'FFFFFFFF)
<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 (or for the specified number of bytes). 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.

DATA_SEARCH

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
:
```

7.2.15	**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 is not performed in the internal I/O area.

DISASSEMBLE

— There are two instructions, PUSH/POP and MOV, that store register contents on the stack or recover them from the stack. However, these two instructions are displayed as the PUSH/POP instruction during disassembly as follows:

- MOV.W Rn, @-R7 is displayed as PUSH.W Rn.
- MOV.W @R7+, Rn is displayed as POP.W Rn.
- MOV.L ERn, @ - ER7 is displayed as PUSH.L ERn.
- MOV.L @ER7+, ERn is displayed as POP. L ERn.

For assembly, of course both instructions can be specified.

— When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, but disassembly display is performed.

Examples

1. To disassemble and display six instructions starting from address H'1000:

```
:DA  1000  @6  (RET)
   ADDR      CODE    LABEL    MNEMONIC    OPERAND
 00001000 7A07000F  !MAIN    MOV.L       #00F0000:32,ER7
          0000
 00001006 79000000           MOV.W       #0000:16,R0
 0000100A 79010000           MOV.W       #0000:16,R1
 0000100E 5E001200           JSR         @!INIT
 00001012 A800               CMP.B       #00:8,R0L
 00001014 586000E8           BNE         !MAIN10
:
```

2. To disassemble and display 16 instructions starting from address H'3000, and to disassemble and display 16 instructions again by only entering (RET):

```
:DA  3000    (RET)
  ADDR      CODE    LABEL    MNEMONIC    OPERAND
 00001000 01006DF6  !SUB     PUSH.L      ER6
 00001004 0D76               MOV.W       R7,R6
 00001006 7900000A           MOV.W       #000A:16,R0
 0000100A 1907               SUB.W       R0,R7
 0000100C 01006DF5           PUSH.L      ER5
 00001010 01006DF4           PUSH.L      ER4
 00001014 79050001           MOV.W       #0001:16,R5
 00001018 78606BA5           MOV.W       R5,@(0000FFF6:32,ER6)
          0000FFF6
 00001020 78606B25           MOV.W       @(0000FFF6:32,ER6),R5
          0000FFF6
 00001028 AD01               CMP.B       #01:8,R5L
 0000102A 79350001           SUB.W       #0001:16,R5
 0000102E 58E02100           BGT         003132:16
 00001032 1956               SUB.W       R5,R6
 00001034 0000               NOP
:(RET)
 00001036 78606BA5           MOV.W       R5,@(0000FFF8:32,ER6)
          0000FFF8
 0000103E 78606BA5           MOV.W       @(0000FFF6:32,ER6),R5
          0000FFF8
    :                         :
    :                         :
```

DISPLAY_COVERAGE		
7.2.16	DISPLAY_COVERAGE DCV	Displays coverage trace results

Command Format

- Display : DISPLAY_COVERAGE[Δ<start address>Δ<end address>] [;<option>[ΔN]] (RET)

<start address>: Start address of coverage trace display
<end address>: End address of coverage trace display
<option>: Display format of coverage trace display
A: Address display
D: Dump display
Default: Address display
N: Displays unexecuted or unaccessed addresses if option A is specified

Description

- Display

— Displays coverage trace information of addresses in the range from <start address> to <end address> (accessed during GO command execution) as address values or in display-dump format. The coverage trace acquisition range is a 2-Mbyte address area. When the MCU has a 64-kbyte or a 1-Mbyte address area, all spaces can be acquired. When the MCU has a 16-Mbyte address area, the trace range is within 2 Mbytes from the base address specified with the EXECUTION_MODE command with the BS option. Refer to section 7.2.19, EXECUTION_MODE. When the address range specification is omitted, all address information to be acquired in the program area is displayed.

— Coverage trace can be executed after emulator system program initiation. Coverage trace settings can be initialized with the SET_COVERAGE command.

— The address display format is as follows:

```
<ADDR>   <ADDR>      <ADDR>   <ADDR>  ...........
xxxxxxxx – xxxxxxxx     xxxxxxxx – xxxxxxxx   ............
   .          .            .          .
   .          .            .          .
   .          .            .          .
```

Displays executed or accessed addresses in one of the following two ways:

xxxxxxxx – xxxxxxxx: Memory area between these two addresses was executed or accessed.
xxxxxxxx: Only this address was accessed.

— If option [;AΔN] is specified, coverage trace information is displayed for addresses in the specified memory range which were not executed or accessed during GO command execution. Although the display format is the same as the address display format, note that the displayed addresses have not been executed or accessed in this case.

— If option D is specified, coverage trace information is displayed in the following display-dump format: <start address> is a multiple of 8 and <end address> is a multiple of 8 – 1.

```
<ADDR>                          <  D  A  T  A  >
 xxxxxxxx        yy       yy       yy       yy       yy       ......................................
   (a)                             (b)
```

(a) Address
(b) Address access information in hexadecimal (00 to FF). Each bit represents one memory address, and an accessed address is indicated as 1.

Example: 00001000 8F 00 ......

The first byte of data 8F indicates that addresses 1000, 1004, 1005, 1006, and 1007 were accessed.

```
Data value      (8)      (F)
Bit pattern     1000     1111
                ↑        ↑
Address        1000     1004
```

DISPLAY_COVERAGE

— Coverage trace information is valid until re-initialization is provided with the SET_COVERAGE command. Coverage trace information of addresses accessed by the user program during GO command execution continues to be acquired until a new coverage trace initialization.

Note

The coverage trace range is 2 Mbytes from the base address specified by the BS option of the EXECUTION_MODE command.

Examples

1. To display coverage trace information for the area from H'400 to H'7FFF:

```
:DCV 400 7FFF  (RET)
  <ADDR>   <ADDR>   <ADDR>   <ADDR>   <ADDR>   <ADDR>
 00000400—00000503 00000700—00000703 00000800-00000815
 00007000—00007103 00007F20—00007FFF
:
```

2. To display addresses of areas not executed by GO command execution:

```
:DCV ;A N  (RET)
  <ADDR>   <ADDR>   <ADDR>   <ADDR>   <ADDR>   <ADDR>
 00000504—000006FF 00000704—000007FF 00000816—00006FFF
 00007104—00007F1F 00008002—00008003 00008006—0000FFFF
:
```

3. To display executed addresses in display-dump format:

```
:DCV ;D  (RET)
 <ADDR>                   <  D  A   T  A >
 00000000   FF C0 F0 00 00 00 00 00  00 00 00 00 00 00 00 00
 00000080   FF FE F0 00 00 00 00 00  FF FF FF 00 00 00 00 00
    :                 :                         :
:
```

7.2.17	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 determined as <end address> or <number of bytes> according to the inequalities given below. Default is 256 bytes, as size.
End address: <start address> ≤ specified value
Number of bytes: <start address> > specified value
<display unit>: Size of display unit in bytes
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 only the (RET) key is pressed 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

The user can execute this command in parallel mode. However, when displaying the user system memory or internal I/O area, the program stops every time 16 bytes are displayed. Therefore, emulation does not operate in realtime. Software standby or sleep state is cancelled, and the execution starts from an instruction following the SLEEP instruction. Emulation memory, internal ROM and internal RAM areas can be emulated in realtime, but the contents are not displayed in software standby or sleep state.

Examples

1. To display a memory dump from addresses H'0 to H'2F:

```
:D 0 2F (RET)
 <ADDR>                 <    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)
 <ADDR>                 <    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 @200 (RET)
 <ADDR>                <    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  "................"
    :               :                   :                     :
 000011F0  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
:(RET)                  Entering (RET) displays next 16 lines.
 00001200  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001210  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001220  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
 00001230  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
    :               :                   :                     :
 000012F0  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  "................"
:
```

END

7.2.18	**END** **E**	**Cancels parallel mode**

Command Format

- Cancellation # END (RET)

Description

- Cancellation

— Cancels parallel mode during GO command execution.

— Entering the END command clears old trace information and starts storing new trace information.

Example

To cancel parallel mode during GO command execution:

```
: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
```

7.2.19	EXECUTION_MODE EM	Specifies and displays execution mode

Command Format

- Setting : EXECUTION_MODE [ΔSTR=<option 1>][ΔBRQ=<option 2>]
  [ΔTIME=<option 3>][ΔPB8=<option 4>]
  [ΔBS=<option 5>][ΔLD=<option 6>]
  [ΔTRG=<option 7>][ΔMON=<option 8>][;C] (RET)

- Display : EXECUTION_MODE (RET)

<option 1>: Specifies whether AS, RD, and WR are output to user system, and whether WAIT can be input at emulation memory access.
E: Outputs AS, RD, and WR, and inputs WAIT when accessing emulation memory
D: Does not output AS, RD, and WR and prohibits input of WAIT when accessing emulation memory (default at emulator shipment)

<option 2>: Specifies whether the BREQ (bus request) signal input is enabled.
E: Enables the BREQ signal input
D: Disables the BREQ signal input (default at emulator shipment)

<option 3>: Specifies the minimum time to be measured by the GO command or PERFORMANCE_ANALYSIS command.
1: 1 µs (default at emulator shipment)
2: 250 ns

<option 4>: Specifies whether external probe 8 is used for a synchronous break on multiple emulators (multibreak).
N: Uses external probe 8 as a normal probe (default at emulator shipment)
M: Uses external probe 8 for multibreak detection

<option 5>: Specifies the coverage trace acquisition range.
0 to F (default at emulator shipment is 0)

<option 6>: Specifies whether the MOV or EEPMOV instruction is executed when memory is accessed with a host-related command.
M: MOV is used
E: EEPMOV is used (default at emulator shipment)

<option 7>: Specifies whether a pulse is output from the trigger output probe of the emulator pod without a break occurring when a hardware break condition is satisfied.
E: Outputs a pulse from the trigger output probe without a break occurring when a hardware break condition is satisfied
D: Breaks when a hardware break condition is satisfied (default at emulator shipment)

<option 8>: Specifies how long the interval is for execution status display.
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 AS (address strobe), RD (read), HWR (higher write), LWR (lower write), and WAIT (wait) input/output at emulation memory access.

(a) To specify that the signals, AS, RD, HWR, and LWR are not output to the user system and the WAIT signal is not input from the user system to the MCU when emulation memory is accessed:

```
: EXECUTION_MODE STR=D (RET)
```

(b) To specify that the signals, AS, RD, HWR, and LWR are output to the user system and the WAIT signal is input from the user system to the MCU when emulation memory is accessed:

```
: EXECUTION_MODE STR=E (RET)
```

The set signal inputs and outputs affect user program execution and emulation memory access with emulator command execution, such as memory modification/reference.

— Enables/disables the BREQ signal (bus request signal) input during realtime emulation.

(a) To disable the BREQ signal input during realtime emulation:

```
: EXECUTION_MODE BRQ=D (RET)
```

(b) To enable the BREQ signal input during realtime emulation:

```
: EXECUTION_MODE BRQ=E (RET)
```

— Specifies the minimum time to be measured by the GO command execution time measurement or by the PERFORMANCE_ANALYSIS command.

(a) To set the minimum time to 1 µs:

```
: EXECUTION_MODE TIME=1 (RET)
```

(b) To set the minimum time to 250 ns:

```
: EXECUTION_MODE TIME=2 (RET)
```

— Sets whether or not external probe 8 is used for multibreak. Refer to the description on external probes in section 5.4.1, Break Function, for details.

(a) To specify that emulator’s external probe 8 is used as a normal probe (the same as probes 1 to 7):

```
: EXECUTION_MODE PB8=N (RET)
```

(b) To specify that emulator’s external probe 8 is used for multibreak detection:

```
: EXECUTION_MODE PB8=M (RET)
```

— Specifies the coverage trace acquisition range. The following 2-Mbyte ranges can be selected by entering data from H'0 to H'F.

Data	Address Range	Data	Address Range
0	H'0 to H'1FFFFF	8	H'800000 to H'9FFFFF
1	H'100000 to H'2FFFFF	9	H'900000 to H'AFFFFF
2	H'200000 to H'3FFFFF	A	H'A00000 to H'BFFFFF
3	H'300000 to H'4FFFFF	B	H'B00000 to H'CFFFFF
4	H'400000 to H'5FFFFF	C	H'C00000 to H'DFFFFF
5	H'500000 to H'6FFFFF	D	H'D00000 to H'EFFFFF
6	H'600000 to H'7FFFFF	E	H'E00000 to H'FFFFFF
7	H'700000 to H'8FFFFF	F	H'F00000 to H'FFFFFF and H'000000 to H'0FFFFF

Example: To set the address range of H'700000 to H'8FFFFF

```
: EXECUTION_MODE BS=7 (RET)
```

EXECUTION_MODE

— Specifies whether the EEPMOV or MOV instruction is used for program loading or saving. When the EEPMOV instruction is used, the program is transferred in byte units and so the transfer speed is high.

(a) To use the MOV instruction:

```
: EXECUTION_MODE LD=M (RET)
```

(b) To use the EEPMOV instruction:

```
: EXECUTION_MODE LD=E (RET)
```

— Specifies whether a pulse is output without stopping emulation by a hardware break specified with the BREAK_CONDITION1,2,3,4,5,6 command. When a hardware break condition is satisfied, a pulse is output to the trigger output probe. For details, refer to section 5.8, Trigger Output. Note that the software break specified with the BREAK or BREAK_SEQUENCE command is always valid regardless of this setting. If TRG = E is specified, the sequential break specified with the GO command becomes invalid and a pulse is output whenever hardware break conditions are satisfied.

(a) To stop the program by a hardware break:

```
: EXECUTION_MODE TRG=D (RET)
```

(b) To output a pulse without stopping the program by a hardware break:

```
: EXECUTION_MODE TRG=E (RET)
```

— Specifies the interval length of PC display during GO command execution.

(a) To disable PC display:

```
: EXECUTION_MODE MON=0 (RET)
```

(b) To enable PC display every 200 ms:

```
: EXECUTION_MODE MON=1 (RET)
```

(c) To enable PC display 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)
  STR=D  BRQ=D  TIME=1uSEC  PB8=N  BS=0  LD=E  TRG=D  MON=1    (Displays current value)
  STR (E: ENABLE/D: DISABLE) ?  E (RET)        (Inputs/outputs AS, RD, LWR, HWR, and WAIT)
  BRQ (E: ENABLE/D: DISABLE) ?  (RET)          (No modification)
  TIME (1: 1uSEC/2: 0.25uSEC) ?  2 (RET)       (Minimum time is 250 ns)
  PB8 (M: MULTI/N: NORMAL) ? . (RET)           (Command is terminated. Modification is valid.)
:
```

Examples

1. To set the coverage trace range 2 Mbytes from address H'200000 and write the setting in the configuration file:

```
:EM BS=2 ;C (RET)
 ***  85: COVERAGE INITIALIZED
 CONFIGURATION WRITE OK (Y/N) ? Y (RET)
:
```

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)
 STR=D  BRQ=D  TIME=1uSEC  PB8=N  BS=0  LD=E  TRG=D  MON=1
 STR (E:ENABLE/D:DISABLE) ? (RET)          (Input (RET) for no modification)
 BRQ (E:ENABLE/D:DISABLE) ? (RET)
 TIME (1:1uSEC/2:0.25uSEC) ? 2 (RET)
 PB8 (M:MULTI/N:NORMAL) ? (RET)
 BS (0 - F) ? .(RET)
:
```

7.2.20	FILL F	Writes data to memory

Command Format

- Write: FILLΔ<start address>(Δ<end address>/Δ@<number of bytes>)[Δ<data>]
  [;[<size>][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
	Default: Verification performed

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 cannot be written to an internal I/O area. If a write to the internal I/O area is attempted, execution terminates just before entering the internal I/O area.
  - — If W or L is specified as <size>, but the start address is odd, the lowest bit is rounded down to the preceding even address. Writing never exceeds the specified <end address>.

Example

To fill the entire area from addresses H'0 to H'6FFF with 1-byte data H'00:

```
:F 0  6FFF 0  (RET)
:
```

GO

7.2.21	**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
<mode>: Emulation mode
S1: Sequential break mode 1
S2: Sequential break mode 2
S3: Sequential break mode 3
R=<n>: Cycle reset mode, n = 1 to 9
I1, I2: Time interval measurement modes 1, 2
N: Invalidates break conditions temporarily
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 MCU, 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 user program is executed in one of the following modes.

(a) Sequential break mode 1 (S1)
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>.

(b) Sequential break mode 2 (S2)
Realtime emulation stops only when break conditions set by the BREAK_CONDITION1,2,3 command are satisfied in the sequence of <condition 3>, <condition 2>, and <condition 1>.

(c) Sequential break mode 3 (S3)
Realtime emulation stops only when break conditions set by the BREAK_CONDITION1,2,3,4 command are satisfied in the sequence of <condition 4>, <condition 3>, <condition 2>, and <condition 1>.

(d) Cycle reset mode (R=n; n=1 to 9)
An RES signal is input to the MCU 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.

(e) Time interval measurement mode 1 (I1)
The execution time from the point when <condition 2> is satisfied until <condition 1> is satisfied is measured. Program execution stops whenever <condition 1> is satisfied.

(f) Time interval measurement mode 2 (I2)
The total execution time from the point when <condition 2> is satisfied until <condition 1> is satisfied is measured. Even if these break conditions are satisfied, the program does not stop and the execution time between BREAK_CONDITION 2,1 is measured. When these conditions are satisfied twice or more, the time is added to the previous measured time.

(g) 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_CONDITION1,2,3,4,5,6 command are invalidated temporarily, and user program 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 1	• Conditions must be specified with the BREAK_CONDITION1,2 command. • Conditions specified with the BREAK_CONDITION3,4,5,6 command are ignored.
Sequential break mode 2	• Conditions must be specified with the BREAK_CONDITION1,2,3 command. • Conditions specified with the BREAK_CONDITION4,5,6 command are ignored.
Sequential break mode 3	• Conditions must be specified with the BREAK_CONDITION1,2,3,4 command. • Conditions specified with the BREAK_CONDITION5,6 command are ignored.
Cycle reset mode	• Software break conditions specified with the BREAK or BREAK_SEQUENCE command and hardware break conditions specified with the BREAK_CONDITION1,2,3,4,5,6 command are ignored. • All conditions specified with the TRACE_CONDITION command are ignored. • Parallel mode cannot be entered. • The TM option cannot be specified.
Time interval measurement mode 1, 2	• Conditions must be specified with the BREAK_CONDITION1,2 command. • Software break conditions specified with the BREAK or BREAK_SEQUENCE command and hardware break conditions specified with the BREAK_CONDITION3,4,5,6 are ignored. • All conditions specified with the TRACE_CONDITION command are ignored. • Parallel mode cannot be entered. • Time measurement specified with the PERFORMANCE_ANALYSIS is not executed.

— 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_CONDITION6 is invalid when a break address is specified.

— Program fetch addresses are displayed at the interval specified with the MON option of the EXECUTION_MODE command during realtime emulation. If the TM option is specified, memory contents of the address specified with the TRACE_MEMORY command are also displayed. Two-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 = zzzz
```

xxxxxxxx: Program fetch address
yyyyyyyy: Address of memory contents to be displayed (valid when the TM option is specified)
zzzz: Memory contents (2 bytes)

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 (the contents before modification are displayed). When specifying the access with DMA, the display contents are not updated until access is completed with DMAC.

— 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  CCR=80:I*******  EXR=07:*****210                  (a)
MACH=00000000   MACL=00000000
ER0 – ER3   00000000  00000001  00000002  00000003
ER4 – ER7   00000008  00000009  0000000A  0000000B
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.
Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and therefore, cannot be displayed.

(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.

— During user program execution, MCU 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 7-13 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 xxxxxxxx	Emulation stopped at a breakpoint xxxxxxxx specified with the BREAK command
STOP ADDRESS	An instruction at the break address was executed with the GO command
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 6	A break condition specified with the BREAK_CONDITION6 command was satisfied
BREAK CONDITION 1,2,3,4,5,6	A break condition was satisfied when the break conditions specified with the BREAK_CONDITION1,2,3,4,5,6 command were satisfied (refer to notes)
BREAK CONDITION S1	Sequential break conditions specified with the BREAK_CONDITION1,2 command were satisfied
BREAK CONDITION S2	Sequential break conditions specified with the BREAK_CONDITION1,2,3 command were satisfied
BREAK CONDITION S3	Sequential break conditions specified with the BREAK_CONDITION1,2,3,4 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 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

Table 7-14 Execution Status Display

Display	Meaning
** RUNNING	Program execution has started. This is displayed only once at GO command execution and parallel mode cancellation. This display disappears when the message shown below (**PC=xxxxxxxx [yyyyyyyy=zzzz]) is displayed during program execution.
**PC=xxxxxxxx [yyyyyyyy=zzzz]	Displays program fetch addresses during execution at the interval specified with the MON option of the EXECUTION_MODE command. If the TM option was specified, each address specified with the TRACE_MEMORY command and its contents are also displayed and updated.
** VCC DOWN	VCC (power supply voltage) is not correct and the emulator forcibly terminates the user program. Provide the correct power supply voltage and re-execute.
** RESET	RES signal is low. The MCU has been reset.
** WAIT A = xxxxxxxx	WAIT signal is low; the value on the address bus is displayed. At the refresh cycle, the address is not displayed.
*** HARDWARE STANDBY	The STBY signal is low but since this signal is not input to the MCU, the MCU does not enter hardware standby state in the emulator.
** SOFTWARE STANDBY	The MCU is in software standby state.
** SLEEP	The MCU is in sleep state.
** BREQ	The BREQ signal is low.
** BACK	The BACK signal is low.
** TOUT A = xxxxxxxx	The AS signal has remained high for 80 µs or more; the value on address bus is displayed.

Notes

1. When the hardware break condition (BREAK CONDITION1,2,3,4,5,6 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 CONDITION1,2,3,4,5,6 (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. If TRG = E is specified as a TRG condition with the EXECUTION_MODE command, a break does not occur even if a break condition specified with the BREAK_CONDITION1,2,3,4,5,6 command is satisfied. In this case, a pulse is output from the trigger output probe pin.

Examples

1. To reset the MCU and start emulation from the reset vector PC address:

```
:G  RESET (RET)
 ** PC=00001130
```

2. To start emulation from address H'1000 and stop emulation just after 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 2:

```
:G ; S2 (RET)
 ** PC=00004250
```

4. To invalidate current software break conditions and hardware break conditions, start emulation, and display the contents of the address specified with the TRACE_MEMORY command:

```
:G ; N TM (RET)
 ** PC=00006642    0000FFD0=0080
```

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
```

HELP		
7.2.22	**HELP** **HE**	**Displays all commands and command format**

Command Format

```
•   Display    :  HELP  (RET)                  (Displays all commands.)
               :  HELP Δ <command> (RET)       (Displays command format.)
```

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                   BI     : BACKGROUND_INTERRUPT
 **B      : BREAK
   BC,BC1,BC2,BC3,BC4,BC5,BC6 : BREAK_CONDITION,1,2,3,4,5,6
 **BS     : BREAK_SEQUENCE             CH     : CHECK
 **CL     : CLOCK                     *CC     : COMMAND_CHAIN
  *CV     : CONVERT                    DC     : DATA_CHANGE
   DS     : DATA_SEARCH               *DA     : DISASSEMBLE
   DCV    : DISPLAY_COVERAGE          *D      : DUMP
  *E      : END                        EM     : EXECUTION_MODE
   F      : FILL                       G      : GO
  *HE     : HELP                      *HT     : HISTORY
  *ID     : ID
  *LED,*LED1,*LED2,*LED3,*LED4 : LED,1,2,3,4
  *LT,*LT1,*LT2 : LED_OUT,1,2          MP     : MAP
  *M      : MEMORY                     MD     : MODE
   MV     : MOVE                       MR     : MOVE_TO_RAM
   PA     : PERFORMANCE_ANALYSIS      *P      : PRINT
   Q      : QUIT                      *RX     : RADIX
   R      : REGISTER                   RS     : RESET
   RT     : RESULT                     SCV    : SET_COVERAGE
  *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
```

HELP

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 the display when the E7000 is used. The display when using the E7000PC is a little different.

2. To display each command format:

```
:HELP GO (RET)
 (Displays GO command format)
:
```

7.2.23	HISTORY HT	Displays input command history

Command Format

- Display : HISTORY (RET) (Displays all input commands)

  : HISTORY <history number> (RET) (Displays the input command of specified history number)

<history number>: History number (1 to 16)

Description

- Display

  — Displays the 16 commands most recently input including the HISTORY command in the input order.

  — If <history number> is entered, the command corresponding to <history number> is displayed as shown below and the emulator enters command input wait state. When the (RET) key is pressed, the displayed command is executed.

Note

Subcommands cannot be displayed by the HISTORY command.

Example

```
:HISTORY (RET)
 01 MAP
 02 MAP  0 1FFFF;S
 03 F 0 1000 FF
 04 B 300
 05 BC1 A=104
 06 HISTORY
:HISTORY 5 (RET)
:BC1 A=104_ ------------Enters command input wait state
```

ID		
7.2.24	**ID** **ID**	**Displays emulator system program version**

Command Format

- Display : ID (RET)

Description

- Display

  Displays the emulator system version and revision numbers.

Example

To display the emulator system version and revision numbers:

```
:ID (RET)
 H8S/2655 E7000 (HS2655EPD70SF) Vn.m
 Copyright (C) Hitachi,LTD. 1995
 Licensed Material of Hitachi, Ltd.
:
```

7.2.25	**LED1,2,3,4** **LED1,2,3,4**	**Specifies, displays, or cancels memory contents display on LEDs**

Command Format

- Specification : LED(1/2/3/4)Δ<address>[;<size>] (RET)
- Initialization : LED[(1/2/3/4)]Δ I (RET)
- Display : LED[(1/2/3/4)] (RET)
- Cancellation : LED[(1/2/3/4)][Δ] – (RET)

<address>: Memory address whose contents are to be displayed on LEDs
<size>: Display data size
B: 1 byte
W: 2 bytes
L: 4 bytes
Default: 1 byte

Description

- Specification

— Specifies the memory addresses whose contents are to be displayed on the LEDs on the optional bus monitor board. After an address is specified with this command, the corresponding data is displayed on the LEDs when the address is accessed by the user program. Up to four addresses can be specified by the LED1 to LED4 commands, and up to four bytes of data can be displayed for one address.

— When specifying W (two bytes) or L (four bytes) as the size, specify a multiple of two or four as the address, respectively. When an odd address is specified, 1-byte (B) data is displayed regardless of the size specification. When L is specified and the address is not a multiple of four, the size is automatically set to W and B, when the address is even and odd, respectively.

— The LEDs turn on, and the display is updated, only after the specified address is accessed by the user program. This applies even when the contents of the address are updated in the emulator as in the case of a timer counter, and neither is the displayed data updated when accessed by an emulator command such as MEMORY.

LED1,2,3,4

- Initialization

  — Turns off the specified LED. The previously specified address remains valid until the address is cancelled or a new address is specified, and its contents are still displayed when the address is accessed by the user program again. When 1, 2, 3, and 4 are omitted, all the LEDs are extinguished.

  - To turn off all the LEDs:

    ```
    : LED I  (RET)
    ```

  - To turn off LED 2:

    ```
    : LED2 I  (RET)
    ```

- Display

  Displays the currently specified addresses. When 1, 2, 3, and 4 are omitted, the addresses specified for all the LEDs are displayed. When no address is specified, a blank will be displayed.

  — To display all specified addresses:

  ```
  : LED  (RET)
   LED1 = xxxxxxxx    y    zzzz...
   LED2 = xxxxxxxx    y    zzzz...
   LED3 = xxxxxxxx    y    zzzz...
   LED4 = xxxxxxxx    y    zzzz...
  ```

  ```
     xxxxxxxx:   Address
            y:   Display size (B: 1 byte, W: 2 bytes, L: 4 bytes)
      zzzz...:   Symbol (only when a symbol is specified)
  ```

  — To display the address specified for LED3:

  ```
  : LED3  (RET)
   LED3 = xxxxxxxx    y    zzzz...
  ```

- Cancellation

  Cancels the specified address. When 1, 2, 3, and 4 are omitted, the addresses specified for all the LEDs are cancelled.

  — To cancel all specified addresses:

  ```
  : LED –  (RET)
  ```

  — To cancel the address specified for LED2:

  ```
  : LED2 –  (RET)
  ```

Examples

1. To display the 2-byte data stored at address H'20000 on LED1:

```
:LED1 20000;W (RET)
:
```

2. To temporarily turn off LED2 (initialization):

```
:LED2 I (RET)
:
```

3. To display all the current address specifications:

```
:LED1 (RET)
 LED1 = 00000882 W
 LED2 = 00000FFE B
 LED3 =
 LED4 =
:
```

4. To cancel the address specified for LED1:

```
:LED1 – (RET)
:
```

LED_OUT1,2		
7.2.26	**LED_OUT1,2** **LT1,2**	**Specifies and displays analog output of LED display data**

Command Format

- Specification : LED_OUT(1/2)Δ<LED number>[;<data position>] (RET)
- Display : LED_OUT(1/2) (RET)

<LED number>: LED number whose displayed data is to be output from the analog terminals

1: LED1 data is output

2: LED2 data is output

<data position>: Position of data to be output from the analog terminals

H: High-order 16 bits of displayed data

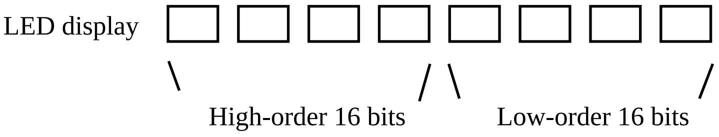
L: Low-order 16 bits of displayed data

Default: Low-order 16 bits of displayed data

Description

- Specification

— Outputs data displayed on LED1 or LED2 on the optional bus monitor board as analog data, from two analog output terminals 1 and 2 located on the monitor board. The output from terminals 1 and 2 is specified by the LED_OUT1 and LED_OUT2 commands, respectively. Either the high-order or low-order 16 bits of data displayed on LED1 or LED2 can be output.



— Analog data is output when the address specified by LED1,2 is accessed by the user program. Accordingly, addresses and LED numbers output by LED1,2 must be specified before program execution. Note that analog data output is undefined until the LED address is accessed by the user program. In addition, analog data output is undefined if the LED is not displayed.

- Display

  Displays the LED number whose display data is being output from the analog terminals, and the data position. When 1 and 2 are omitted, the specifications for both LED_OUT1 and LED_OUT2 are displayed.

  — To display specifications for both the LED_OUT1 and LED_OUT2:

```
: LED_OUT  (RET)
  LED_OUT1  LEDn m
  LED_OUT2  LEDn m
```

  LEDn: LED number whose displayed data is being output
  m: Output data position
  H: High-order 16 bits
  L: Low-order 16 bits

  — To display specifications for LED_OUT2:

```
: LED_OUT2  (RET)
  LED_OUT2  LEDn m
```

Examples

1. To output the high-order 16 bits of the data displayed by LED2 from analog output terminal 1:

```
:LT2 2 H (RET)
:
```

2. To display the specifications for analog output terminals 1 and 2:

```
:LT (RET)
 LED_OUT1 LED1 L
 LED_OUT2 LED2 L
:
```

7.2.27	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
S: Emulation memory in the emulator pod
SW: Emulation memory with write protection in the emulator pod
SG: Emulation memory with access inhibition in the emulator pod
E: Optional memory board
EW: Optional memory board with write protection
EG: Optional memory board with access inhibition
W: Read-only memory (write-protected) access
G: Guarded memory area (access-inhibited)

Description

- Specification

— Allocates standard emulation memory or optional emulation memory board

(a) Standard emulation memory allocation

Allocates standard emulation memory (512 kbytes) in the emulator pod 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.

Note: When the MCU has only 64 kbytes of memory space, the start and end addresses must be H'0 and H'FFFF, respectively. (In this case, however, remaining memory space is reduced by 128 kbytes.)

```
: MAP  0  1FFFF;S  (RET)
```

(b) Memory board (option) emulation memory allocation

Memory board (option) is allocated in 1-Mbyte units. The emulation memory can be write-protected or access-inhibited by specifying EW or EG, respectively. The start address is rounded down to 0 or a multiple of H'100000, and the end address is rounded up to a multiple of H'100000, minus one.

Note: The emulation memory area allocated on the memory board is accessed in 3-state 1-wait cycle. Therefore, memory board emulation memory must not be assigned to a 2-state access area. If assigned, the area cannot be accessed correctly. In addition, write accesses to the memory board emulation memory cannot be executed correctly when the write data buffer function is used. Do not use the write data buffer function when emulation memory is allocated on the memory board.

```
: MAP  0  FFFFF;E  (RET)
```

Both emulation memories can be accessed in realtime in parallel mode. To move the memory to the user system, specify option U. After allocation, the size of the unused emulation memory is displayed.

```
REMAINS EMULATION MEMORY S=xxxxx/E=xxxxxx
          S = xxxxx  (Standard emulation memory)
          E = xxxxxx  (Optional emulation memory)
```

— One write-protected area and one access-inhibited area can be allocated 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 inhibited: Program execution stops when the area is accessed (read/written) by the user program

With an emulator command, the user can read from and write to the write-protected area. In single chip mode, the UNUSABLE area has already been specified as access-inhibited at start up.

MAP

- 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;x                                                              (a)
 INTERNAL ROM  = xxxxxxxx-xxxxxxxx                                                (b)
 RESERVED AREA  = xxxxxxxx-xxxxxxxx  xxxxxxxx-xxxxxxxx                            (c)
 INTERNAL RAM  = xxxxxxxx-xxxxxxxx                                                (d)
 INTERNAL I/O      = xxxxxxxx-xxxxxxxx                                            (e)
 GUARDED AREA = xxxxxxxx-xxxxxxxx    WRITE PROTECT AREA= xxxxxxxx-xxxxxxxx       (f)
 REMAINS EMULATION MEMORY    S=xxxxx/E=xxxxxx                                     (g)
```

(a) Address range and memory attributes
(b) Internal ROM address range (displayed only when the MCU has an internal ROM)
(c) Reserved area range
(d) Internal RAM address range
(e) Internal I/O address range
(f) Guarded memory area and write-protected area address ranges (displayed only when they are specified)
(g) Unused emulation memory size in hexadecimal
S=xxxxx (Standard emulation memory)
E=xxxxxx (Optional 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 – ;W (RET)   Cancels all write protection specifications.

: MAP – ;G (RET)   Cancels all access inhibition specifications.

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. Emulation memory cannot be allocated to the internal ROM, internal RAM, and internal I/O areas. These areas cannot be specified as a write-protected or access-inhibited area.
3. Optional memory (E) should not be allocated to a 2-state access area; if it is, it may not be accessed correctly. Standard memory (S) in the emulator pod must be assigned to 2-state access areas.
4. Emulation memory may be temporarily assigned to the reserved area or an unusable area in single chip mode. However, in the actual MCU, note that an unusable area cannot be accessed.
5. When optional memory (E) is allocated, the write data buffer functions must not be used. If the function is used, the memory cannot be accessed correctly.

Examples

1. To allocate standard emulation memory in the emulator pod to the address range from H'0 to H'1FFFF:

```
:MP 0 1FFFF;S (RET)
 REMAINS EMULATION MEMORY  S=60000/E=000000
:
```

2. To make the address range from H'40000 to H'5FFFF for write protection:

```
:MP 40000 5FFFF ;W (RET)
:
```

MEMORY		
7.2.28	**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 <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 are 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. When the user system memory or internal I/O area contents are displayed, the program stops at every display or when writing is performed. The emulator therefore, does not operate in realtime. Software standby and sleep states are cancelled and program starts to execute from the next instruction of the SLEEP instruction. In the emulation memory, internal ROM, and internal RAM, the emulator operates in realtime. Their contents are not displayed in software standby or sleep state.

2. By controlling the EAE bit of the bus control register L (BCRL), the emulator enables access to all internal ROM areas. Therefore, even if the EAE bit is modified with this command, access can be performed to any internal ROM area with an emulator command.

MEMORY

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)
:
```

7.2.29	**MODE** **MD**	**Specifies or displays MCU operating mode**

Command Format

- Specification : MODE;C (RET)
- Display : MODE (RET)

Description

- Specification

— Interactively specifies the MCU operating mode as shown below.

```
:MODE;C (RET)
 CPU MODE = H8S/2600    OPERATION MODE = 7 (MD2-0=7)    MODE SET = E7000
 PIN MODE = 120/128   INTERNAL ROM SIZE = 128KB   INTERNAL RAM SIZE = 4KB    (1)
 ADC MODE = HI SPEED    DMAC MODE = ENABLE    REFRESH MODE = ENABLE
 SCI CHANEL = 3CH    16BIT TIMER MODE =TPU0-2
 CPU MODE (1:H8S/2000 , 2:H8S/2600) a ?  (2) (RET)
 OPERATION MODE (MD2-0) b ?  (3) (RET)
 MODE SET (1:E7000 MODE , 2:USER MODE) c ?  (4) (RET)
 PIN MODE (1:80/84-A  , 2:100-A  , 3:112  , 4:120/128  , 5:80/84-B  , 6:100-B) d ?  (5) (RET)
 INTERNAL ROM SIZE (1:0KB , 2:32KB , 3:64KB , 4:96KB , 5:128K , 6:256KB) e ?  (6) (RET)
 INTERNAL RAM SIZE (1:1KB  , 2:2KB  , 3:3KB  , 4:4KB  , 5:6KB  , 6:8KB) f ?  (7) (RET)
 ADC MODE (1:NORMAL , 2:HI SPEED) g ?  (8) (RET)
 DMAC MODE (1:DISABLE , 2:ENABLE) h ?  (9) (RET)
 REFRESH MODE (1:DISABLE , 2:ENABLE) i ?  (10) (RET)
 SCI CHANEL (1:2CH , 2:3CH) j ?  (11) (RET)
 16BIT TIMER MODE (1:TPU0-2 , 2:TPU0-5) k ?  (12) (RET)
 CONFIGURATION WRITE OK (Y/N) l ?  (13) (RET)
```

(1) Current settings.

(2) MCU type. a is the current MCU type. (default at emulator shipment is H8S/2600)
Example: Input 2 to select the H8S/2655 series.

(3) MCU operating mode setting. b is the current operating mode. (default at emulator shipment is 7)
Example: Input 6 to select operating mode 6.

(4) Operating mode setting selection. c is the current method.
1: Selects the operating mode set at (3). (default at emulator shipment)
2: Selects the operating mode specified by the mode selection pins (MD2 to MD0) on the user system.

(5) Number of MCU pins. d is the current number of pins. (default at emulator shipment is 120/128)
 Example: Input 3 to select 112 pins.

(6) Internal ROM size. e is the current size. (default at emulator shipment is 128 kbytes)
 Example: Input 2 to select 32 kbytes.

(7) Internal RAM size. f is the current size. (default at emulator shipment is 4 kbytes)
 Example: Input 6 to select 8 kbytes.

(8) Internal A/D converter. g is the current converter.
 1: Selects the converter whose conversion period is 134 states. (default at emulator shipment)
 2: Selects the converter whose conversion period is 20 states.

(9) Internal DMAC selection. h is the current setting.
 1: Disables internal DMAC.
 2: Enables internal DMAC. (default at emulator shipment)

(10) Internal refresh controller selection. i is the current setting.
 1: Disables internal refresh controller.
 2: Enables internal refresh controller. (default at emulator shipment)

(11) Number of internal SCI channels. j is the current number of channels.
 1: 2 channels
 2: 3 channels (default at emulator shipment)

(12) Combination of 16-bit timers. k is the current combination.
 1: TPU0 to TPU2 are used. (default at emulator shipment)
 2: TPU0 to TPU5 are used.

(13) Verification message of specification
 Input Y to perform specification and write the setting to the configuration file, and N to cancel specification without writing the setting.

— Since the settings are written to the configuration file when the E7000 is used, a writable system disk must be set up before MODE command execution. After the MCU type and operating mode are recorded in the configuration file, the E7000 can be initiated as the specified MCU type and 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.

- Display

Displays the MCU, operating mode, and operating mode selection pin (MD0 to MD2) status on the user system in the following format:

```
:MODE (RET)
 CPU MODE = H8S/2600  OPERATION MODE = 7 (MD2-0=7)  MODE SET = E7000
               (a)                      (b)      (c)             (d)
 PIN MODE = 112  INTERNAL ROM SIZE = 128KB  INTERNAL RAM SIZE = 4KB
             (e)                      (f)                        (g)
 ADC MODE = HI SPEED  DMAC MODE = ENABLE  REFRESH MODE = ENABLE
               (h)                    (i)                       (j)
 SCI CHANEL = 3CH  16BIT TIMER MODE = TPU0-2
               (k)                  (l)
```

(a) MCU type
(b) Operating mode
(c) MD0 to MD2 pin status on the user system (Refer to table 7-16.)
(d) Operating mode setting selection
(e) Number of pins
(f) Internal ROM area size
(g) Internal RAM area size
(h) A/D converter function
(i) DMAC selection
(j) Refresh controller selection
(k) Number of SCI channels
(l) 16-bit timer combination

Table 7-16 Operating Mode Selection Pin Status and Display

MD2	MD1	MD0	Display (n)
Low	Low	Low	0
Low	Low	High	1
Low	High	Low	2
Low	High	High	3
High	Low	Low	4
High	Low	High	5
High	High	Low	6
High	High	High	7

Notes

1. When the operating mode setting selection is specified as 1 (operating mode set by MODE command), the operating mode in the emulator is specified by the MODE command, regardless of the operating mode selection pin (MD0 to MD2) status on the user system.

2. When the emulator is not connected to the user system, the operating mode is specified by the MODE command even when the operating mode setting selection is specified as 2 (operating mode set by mode selection pins (MD0 to MD2).

3. When the internal ROM size is set to 128 kbytes, the 128-kbyte internal ROM can be accessed by commands regardless of the EAE bit status. For example, even if the EAE bit is modified with the MEMORY command to make the area from H'010000 to H'01FFFF to be an external address area, the internal ROM is accessed when the area is accessed with emulator commands. However, user program execution starts with the EAE bit status specified by the MEMORY command. In the same way, when the reset signal is input, the internal ROM is accessed if the area is accessed with emulator commands, and user program execution starts with the EAE bit initialized. For details on the EAE bit, refer to the MCU hardware manual.

4. When H8S/2000 is selected as the MCU, the MACH and MACL registers do not exist. Note the following:

   — An error will occur when the MACH or MACL is selected by the .<register> command.
   — The MACH and MACL are omitted from the display with the REGISTER command or the execution result display.
   — The MACH and MACL can be input and displayed with the ASSEMBLE and DISASSEMBLE commands. They can be displayed also with the trace and step functions.

Examples

1. To specify the current MCU type as H8S/2600, the operating mode as mode 5, the operating mode selection as USER, the number of pins as 144, the internal ROM size as 128 kbytes, the internal RAM size as 8 kbytes, the A/D converter as high-speed mode, the DMAC and refresh controller as enabled, the SCI as 3-channel multiprocessor communication/smart card interface support, and the 16-bit timer as TPU0 to TPU5:

```
:MD;C (RET)
 CPU MODE = H8S/2600  OPERATION MODE = 7 (MD2-0=5)  MODE SET = E7000
 PIN MODE = 120/128  INTERNAL ROM SIZE = 32KB  INTERNAL RAM SIZE = 2KB
 ADC MODE = HI SPEED  DMAC MODE = ENABLE  REFRESH MODE = DISABLE
 SCI MODE = NORMAL  SCI CHANEL = 2CH  16BIT TIMER MODE = TPU0-2
 CPU MODE (1:H8S/2600 , 2:H8S/2000) 1 ?  2 (RET)
 OPERATION MODE (MD2-0) 7 ? 5 (RET)
 MODE SET (1:E7000 MODE , 2:USER MODE) 1 ? 2 (RET)
 PIN MODE (1:80/84-A , 2:100-A , 3:112 , 4:120/128 , 5:80/84-B , 6:100-B) 1 ? 3 (RET)
 INTERNAL ROM SIZE (1:0KB , 2:32KB , 3:64KB , 4:96KB , 5:128KB , 6:256KB) 2 ? 5 (RET)
 INTERNAL RAM SIZE (1:1KB , 2:2KB , 3:3KB , 4:4KB , 5:6KB , 6:8KB) 2 ? 6 (RET)
 ADC MODE (1:NORMAL , 2:HI SPEED) 2 ? 2 (RET)
 DMAC MODE (1:DISABLE , 2:ENABLE) 2 ? 2 (RET)
 REFRESH MODE (1:DISABLE , 2:ENABLE) 1 ? 2 (RET)
 SCI CHANEL (1:2CH , 2:3CH) 1 ? 2 (RET)
 16BIT TIMER MODE (1:TPU0-2 , 2:TPU0-5) 1 ? 2 (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 restarted by S)
```

2. To display the current MCU type, operating mode, and operating mode selection pin (MD0 to MD2) status:

```
:MD (RET)
 CPU MODE = H8S/2600  OPERATION MODE = 5 (MD2-0=5)  MODE SET = USER
 PIN MODE = 112  INTERNAL ROM SIZE = 128KB  INTERNAL RAM SIZE = 8KB
 ADC MODE = HI SPEED  DMAC MODE = DISABLE  REFRESH MODE = DISABLE
 SCI CHANEL = 3CH  16BIT TIMER MODE = TPU0-5
:
```

7.2.30	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 the error
yy 'y':	Write data (hexadecimal and ASCII characters)
zz 'z':	Read data (hexadecimal and ASCII characters)

If data is to be written to an area including the internal I/O area, the following warning message is displayed:

```
*** 86:INTERNAL I/O AREA
```

In this case, data is written to addresses other than the internal I/O area.

Example

To transfer data in the address range from H'101C to H'10FC to address H'1000:

```
:MV 101C 10FC 1000  (RET)
:
```

7.2.31	**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>][Δ<access state>]] (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 emulation memory to be allocated
S: Standard emulation memory installed in emulator pod
SW: Standard emulation memory with write protection installed in emulator pod
E: Optional memory board
EW: Optional memory board with write protection
Default: Standard emulation memory installed in 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 where data can be modified. Data is transferred to standard emulation memory and optional emulation memory is performed in 128-kbyte and 1-Mbyte units, respectively. After data transfer, the unused standard emulation memory area is displayed as follows:

```
REMAINS EMULATION MEMORY        S=xxxxx/E=xxxxxx

            S=xxxxx  (Standard emulation memory)

            E=xxxxxx  (Optional emulation memory)
```

— If the internal ROM or I/O area is included in the specified transfer area, data transfer is performed in areas other than these areas.

Note

This function cannot be used in single chip mode.

MOVE_TO_RAM

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=60000/E=000000
:
```

7.2.32	PERFORMANCE_ANALYSIS PA	Specifies, cancels, initializes, or displays performance measurement data

Command Format

- Specification : PERFORMANCE_ANALYSISΔ<subroutine name>Δ<start address> Δ<end address> (RET)
- Cancellation : PERFORMANCE_ANALYSIS[Δ]–[<subroutine name>] (RET)
- Initialization : PERFORMANCE_ANALYSISΔI (RET)
- Display : PERFORMANCE_ANALYSIS[ΔV] (RET)

<subroutine name>: Name of the subroutine whose execution performance is to be measured. Any name can be specified other than I or V.
<start address>: Subroutine entry address
<end address>: Subroutine exit address
I: Specifies initialization of execution performance measurement information
V: Requests display of performance results such as execution time, number of times executed, and total run-time. If V is omitted, execution time rate is displayed in graph form.

Description

- Specification

— Measures the execution time of the subroutine defined by the <start address> and <end address> parameters during user program execution initiated with the GO command. Measurement starts when an address within the range from the start address to the end address is prefetched, halts when an address outside the specified range is prefetched, and restarts when an address within the specified range is prefetched again. The number of times these subroutines are executed can also be counted. The subroutine execution count is incremented every time the subroutine's end address is passed.

— While the measurement and execution count are performed in realtime, please note the following:

Notes: 1. This command measures execution time and counts during program prefetch. While executing the preceding subroutine (whose address has a lower absolute value), the subroutine to be measured is fetched. Interrupt processing starts immediately after that and execution time measurement and counting begin even if the program is not yet being executed.
2. Execution count is doubled when the measured subroutine is in the 8-bit data bus access area.

- Up to four subroutines can be specified.

- The PERFORMANCE_ANALYSIS command cannot be executed during program execution of the STEP or STEP_OVER command.

- Cancellation

- Cancels execution performance measurement corresponding to the specified subroutine.

- If the subroutine name is omitted, all subroutines for execution performance measurement are cancelled.

- Initialization

Clears the current execution time and execution count for all subroutines, as well as the total run-time. The total run-time begins to be measured only after the subroutines measured by this command are assigned. If no subroutines are assigned, the total run-time is not measured.

- Display

Displays execution performance measurement results, in either of the following two formats:

- Execution time ratio displayed in graph form. (Option V is not specified.)

```
: PERFORMANCE_ANALYSIS  (RET)

NAME  S-ADDR    E-ADDR    RATE     0----------------------50----------------------100
SUBA  00000100  00001FF3  D'10.0%  ****
SUBB  00000204  0000215F  D'20.0%  ********
SUBC  00003000  00003457  D'15.0%  ******
 (a)     (b)       (c)      (d)      (e)
-----------------------------------------------------------------------------------------------------
TOTAL RUN-TIME = D'0000H:10M:00S:000020US[:250NS]          (f)
```

(a) Subroutine name (the 8 characters at the head of the subroutine name)
(b) Subroutine start address
(c) Subroutine end address
(d) Execution time ratio as a percentage
(e) Execution time ratio in graph form (in units of 2.5%/asterisk, rounded up)
(f) Total run-time displayed as H (hour), M (minutes), S (second), US (microsecond), and NS (nanosecond). However, when minimum measurement time is specified as 1 µs by the TIME option of the EXECUTION_MODE command, NS display is not available. For details, refer to section 7.2.17, EXECUTION_MODE.

— Execution time and number of executions are displayed as numerical value. (Option V is specified.)

```
: PERFORMANCE_ANALYSIS V  (RET)

NAME   S-ADDR     E-ADDR     RATE      RUN-TIME                                E-COUNT
SUBA   00000100   00001FF0   D'10.0%   D'0000H:00M:05S:001000US[:250NS]  D'00005
SUBB   00002030   0000215F   D'20.0%   D'0000H:00M:10S:010305US[:500NS]  D'00010
  (a)      (b)        (c)       (d)        (e)                                   (f)
--------------------------------------------------------------------------------------------------------
TOTAL RUN-TIME = D'0001H:00M:50S:000020US[:250NS]            (g)
```

(a) Subroutine name (the 8 characters at the head of the subroutine name)
(b) Subroutine start address
(c) Subroutine end address
(d) Execution time ratio
(e) Execution time
(f) Number of executions (If 65,535 is exceeded, ***** is displayed)
(g) Total run-time

Note

If the minimum measurement time is specified by the TIME option of the EXECUTION_MODE command as 1 µs, the maximum measurable time is approximately 305 hours. Alternatively, if it is specified as 250 ns, it is approximately 76 hours.

Examples

1. To measure the execution time of subroutines SUBD (H'5204 to H'5E00) and SUBE (H'787E to H'7EF0) and to initialize the measurement data:

```
:PA SUBD 5204 5E00  (RET)
:PA SUBE 787E 7EF0  (RET)
:PA I  (RET)
:
```

2. To display execution time ratio in graph form:

```
:PA  (RET)

 NAME  S-ADDR    E-ADDR    RATE    0------------50------------100
 SUBD  00005204  00005E00  D'10.0% ****
 SUBE  0000787E  00007EF0  D'20.0% ********
 ---------------------------------------------------------------
 TOTAL RUN-TIME = D'0000H:00M:50S:000020US
```

3. To display execution time and number of executions in numerical form:

```
:PA V  (RET)
 NAME   S-ADDR     E-ADDR     RATE     RUN-TIME                    E-COUNT
 SUBA   00000100   00001FF0   D'10.0%  D'0000H:00M:05S:001000US    D'04024
 SUBB   00002030   0000215F   D'20.0%  D'0000H:00M:10S:010305US    D'09566
 ---------------------------------------------------------------
 TOTAL RUN-TIME = D'0001H:00M:50S:000020US
```

4. To cancel all registered subroutines:

```
:PA -  (RET)
:
```

7.2.33	**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 output
```

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 is not output to the printer or file:

- Program counter during GO command execution
- Help messages for each command (HELP <command name>)
- 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, Emulation Support Function, 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)
:
```

7.2.34	**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
```

QUIT

— 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 QUIT Command

Item	Description
Software breakpoints	Information set by the BREAK and BREAK_SEQUENCE commands
Hardware break conditions	Information set by the BREAK_CONDITION1,2,3,4,5,6 command
Trace condition	Information set by the TRACE_CONDITION, TRACE_MODE, and TRACE_MEMORY commands
Memory map	Information set by the MAP command
Performance analysis information	Information set by the PERFORMANCE_ANALYSIS command
Coverage information	Coverage acquisition status
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)
LED display information	Information set by LED1,2,3,4 and LED_OUT1,2 commands
User interrupt accept in command wait state	Information set by BACKGROUND_INTERRUPT command

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 on the IBM PC. 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) ?  _
```

RADIX

7.2.35	**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 initiation. Numbers may be entered in any radix at any time, provided that each value is prefixed with the appropriate character.

Table 7-18 Radix 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:

```
:RX  (RET)
 RADIX = D:DECIMAL
:
```

REGISTER

7.2.36	**REGISTER** **R**	**Displays register contents**

Command Format

- Display : REGISTER (RET)

Description

- Display

  Displays all register contents.

  Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and therefore, cannot be displayed. However, instruction mnemonic display is performed including the MACH and MACL registers.

Example

To display all register contents:

```
:R  (RET)
 PC=00001000  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000001 00000002 00000003
 ER4 - ER7  00000008 00000009 0000000A 0000F000
:
```

7.2.37	**RESET** **RS**	**Resets MCU**

Command Format

- Reset : RESET (RET)

Description

- Reset

  Resets the MCU. The MCU general-purpose and control register contents will be reset to the following values:

ER0 to ER6:	The value before reset
ER7 (SP):	The value before reset
PC:	Reset vector value
CCR:	H'80
EXR:	H'07
MACH:	The value before reset
MACL:	The value before reset

  The internal I/O register contents will also be reset. However, by controlling the EAE bit of the bus control register L (BCRL), the emulator enables access to all internal ROM areas.

Example

To reset the MCU:

```
:RS  (RET)
 ** RESET IN BY E7000 !
:
```

RESULT		
7.2.38	**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   CCR=80:I*******    EXR=07:*****210                        (a)
 -MACH=00000000    MACL=00000000
 -ER0 - ER3 00000000  00000001  00000002  00000003
 -ER4 - ER7 00000008  00000009  0000000A  0000000B
  RUN-TIME=D'000H:00M:00S:000000US[:000NS]                                (b)
  +++: <cause of termination >                                            (c)
  [LINE NO=<line number symbol>+n]                                        (d)
```

(a) Register contents at program termination
Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and therefore, cannot be displayed.

(b) Execution time

(c) Cause of termination. For a list of these causes, refer to section 7.2.21, GO, section 7.2.42, STEP, and section 7.2.44, STEP_OVER.

(d) If line number symbols are registered, the stop address is displayed as the nearest <line number symbol> + n.

Note

Displayed register contents show values at program termination, not the current values.

Example

To display execution results:

```
:RT  (RET)
 -PC=00001000 CCR=80:I******* EXR=07:*****210
 -MACH=00000000 MACL=00000000
 -ER0-ER3  00000000 00000001 00000002 00000003
 -ER4-ER7  00000008 00000009 0000000A 0000000B
  RUN-TIME=D'000H:00M:00S:000124US
  +++:BREAK POINT   002FF0
:
```

7.2.39	**SET_COVERAGE** **SCV**	**Initializes the coverage trace function**

Command Format

- Initialization : SET_COVERAGE (RET)

Description

- Initialization

  — Initializes the coverage buffer, which acquires address access information during GO command execution. The coverage trace information begins to be acquired from the moment the emulator system is activated.

  The coverage trace range is within a 2-Mbyte memory space. When the MCU's memory space is 64 kbytes or 1 Mbyte, the coverage trace function for the whole area is acquired. However, if the memory space is 16 Mbytes, the coverage trace range is only a 2-Mbyte memory space, starting from the base address specified with the BS option of the EXECUTION_MODE command. Refer to section 7.2.19, EXECUTION_MODE.

  — Coverage trace information is displayed with the DISPLAY_COVERAGE command.

Example

To initialize the coverage buffer:

```
:SCV (RET)
:
```

7.2.40	**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.

SHORT_SYMBOL

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/sub/cmdo  (RET)
:SS \3=msym/sub/comd7  (RET)
:B  !\2, !\3  (RET)
```

2. To display all current short formats:

```
:SS  (RET)
\1 = msym/sub/cmb1
\2 = msym/sub/cmd2
\3 =
\4 =
\5 =
:
```

7.2.41	STATUS ST	Displays emulator execution status

Command Format

- Display : STATUS (RET)

Description

- Display

Displays emulator execution status in the following format:

```
RADIX=(a)     BREAK=(b)     B_CND=(c)     T_CND=(d)
HOST=(e)    SYM=(f)             LINE_SYM=(g)              STEP_INFO=REG:(h) /A:(i)
CLOCK=(j)   PRINT=(k)                        BASE=(l)    EML_MEM=S:(m)/E:(n)
```

(a) RADIX=xxx: Default input number type

BIN: Binary
OCT: Octal
DEC: Decimal
HEX: Hexadecimal

(b) BREAK=D'xxx: Number of breakpoints (decimal)

(c) B_CND=xxxxxx: BREAK_CONDITION1,2,3,4,5,6 command setting. If specified, the specified number is displayed; otherwise, a blank is displayed.

(d) 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.

(e) 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

(f) SYM=D'xxxxx: Number of defined symbols (decimal)

(g) LINE_SYM=D'xxxxx: Number of defined line number symbols (decimal)

(h) STEP_INFO=REG:x_1 x_2 : Register information provided by the STEP command

x_1	1:	Control register (PC, CCR, EXR, MACH, and MACL) information is displayed.
	Space:	No control register (PC, CCR, EXR, MACH, and MACL) information is displayed.
x_2	2:	General-purpose register (ER0 to ER7) information is displayed.
	Space:	No general-purpose register (ER0 to ER7) information is displayed.

(i) /A:xxxxxxxx-xxxxxxxx: Memory address displayed with the STEP command

(j) CLOCK=xx: Clock signal type

8 MHz:	8-MHz emulator internal clock signal
20 MHz:	20-MHz emulator internal clock signal
USER:	User system clock signal
X'TAL:	Crystal oscillator clock signal of the emulator pod

(k) 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.

(l) BASE=x: Indicates the coverage trace address range specified by the BS option of the EXECUTION_MODE command

x1 = 0 to F
0: 000000 to 1FFFFF
1: 100000 to 2FFFFF
2: 200000 to 3FFFFF
:
E: E00000 to FFFFFF
F: F00000 to FFFFFF and 000000 to 0FFFFF

(m), (n) EML_MEM=S:xxxxKB/E:yyyyKB: Indicates the remaining size in the standard emulation memory and optional memory boards.

xxxx: Remaining size in standard emulation memory
yyyy: Remaining size in optional memory boards

Example

To display the emulator status:

```
:ST  (RET)
 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=20MHz   PRINT=X1            BASE=0    EML_MEM=S:0512KB/E:0000KB
```

7.2.42	STEP S	Performs single-step execution

Command Format

- Single step : STEP [Δ<number of execution steps>[Δ<start PC>]]
  [;[<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 PC>: Start address of single-step execution. Default is the current PC address.

<stop PC>: PC address when single-step execution is terminated. Default is <number of execution steps>.

<display option>: Specification of instructions to be displayed

J: Displays 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

- Single step

— Performs single-step execution from specified PC. 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:

Description

```
(a)  PC=00001000     CCR=80:I*******   EXR=07:*****210
     MACH=00000000   MACL=00000000
     ER0 - ER3   00000000 00000001 00000002 00000003
     ER4 - ER7   00000008 00000009 0000000A 0000000B
(b)  <address> <instruction mnemonic>
(c)  MEMORY
       <memory contents>
(d)  +++:<cause of termination>
```

(a) Register information

Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and therefore, cannot be displayed. However, instruction mnemonic display is performed including the MACH and MACL registers.

(b) Address and mnemonic of the instruction that was executed

(c) Memory contents display

(d) Cause of termination

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 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 CONDITION1,2,3,4	Break conditions specified with the BREAK_CONDITION1,2,3,4 command were satisfied.
GUARDED AREA ACCESSED	A guarded memory area was accessed.
WRITE PROTECT	A write-protected area was written to.
ILLEGAL INSTRUCTION	An invalid 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 PC>]  (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 PC>]]; <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 PC>]];[<stop PC>] J  (RET)
```

The following instructions are valid when the J option is specified:

Bcc, BRA, BRN, JMP, BSR, and JSR

— 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. This function is similar to the STEP_OVER command function. If the instruction is in RAM (including internal ROM, RAM, and emulation memory), use the STEP_OVER command whose execution time is shorter.

```
: STEP [<number of execution steps>  [<start PC>]];[<stop PC>] R  (RET)
```

— 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.

— <stop PC> must be the first address of an instruction. If not, execution does not stop.

Notes

1. The step function uses hardware break (BREAK_CONDITION5,6), and therefore BREAK_CONDITION5,6 settings are ignored during STEP command execution.

2. Software breakpoints (including those specified by the BREAK and BREAK_SEQUENCE commands) are ignored during single-step execution.

3. If one of BREAK_CONDITION1,2,3,4 conditions is satisfied, STEP execution may be terminated without executing instructions.

Examples

1. To execute a program one step at a time, starting from the address given by the current PC:

```
:S (RET)
 PC=00001002  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000002 00000003
 ER4 - ER7  00000008 00000009 0000000A 0000FF00
 00001000                 MOV.W     R0,R1
 +++;STEP NORMAL END
:(RET)
 PC=00001004  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000002 00000003
 ER4 - ER7  00000001 00000009 0000000A 0000FF00
 00001002                 MOV.W     #0001:16,R4
 +++;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  CCR=84:I*******   EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000044 000000FF
 ER4 - ER7  00000008 00020000 0000000A 0000FF00
 00001064                 JMP       @00106A:16

 PC=00001072  CCR=80:I*******   EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000044 000000FF
 ER4 - ER7  00000008 00020000 0000000A 0000FF00
 00001070                 NOP
 +++;STOP ADDRESS
:
```

7.2.43	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, CCR, EXR, MACH, and MACL
2: Displays ER0 to ER7
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.

STEP_INFORMATION

```
(a)  PC=00005C60   CCR=80:I*******  EXR=07:*****210
     MACH=00000000  MACL=00000000
(b)  ER0 - ER3  00000000  00000001  00000002  00000003
     ER4 - ER7  00000008  00000009  0000000A  0000000B
(c)  0000F0F2                NOP
(d)  MEMORY
     0000FF80  00  04  00  FF  F0  00  02  00   10  00  02  00  0F  00  00  00  "................"
```

(a) Control register information (PC, CCR, EXR, MACH, and MACL)

Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and therefore, cannot be displayed. However, instruction mnemonic display is performed including the MACH and MACL registers.

(b) General-purpose register information (ER0 to ER7)

(c) Address and assembler mnemonic of executed instruction

(d) Memory contents display

- Display

Displays STEP information according to the specified contents. However, the address and assembler instruction mnemonic of each executed instruction are not displayed.

Examples

1. To display only the contents of control registers (PC, CCR, EXR, MACH, and MACL) during STEP or STEP_OVER command execution:

```
:SI  1 (RET)
:
```

2. To display no register information during STEP or STEP_OVER command execution:

```
:SI - (RET)
:
```

3. To display memory contents from address H'FB80 to H'FB87 during STEP or STEP_OVER command execution:

```
:SI  A=FB80  FB87  (RET)
:
```

4. To display contents according to the specified display information:

```
:SI (RET)
 PC=00005C60  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  0000000 00000001 00000002 00000003
 ER4 - ER7  0000008 00000009 0000000A 0000000B
 MEMORY
 0000FB80  00 04 00 FF F0 00 02 00              "..............""
:
```

7.2.44	**STEP_OVER** **SO**	**Performs single-step execution except for subroutines**

Command Format

- Execution : STEP_OVER [<start PC>][;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 TRAPA instruction beginning at <start address>. If BSR, JSR, or TRAPA is executed, acts as if the subroutine called by the BSR, JSR, or TRAPA is a single instruction. If an instruction other than BSR, JSR, or TRAPA is executed, register contents and the executed instruction are shown after each instruction is executed as in the STEP command.

  — If BSR, JSR, or TRAPA is executed, the STEP_OVER command replaces the instruction following the BSR, JSR, or TRAPA with an instruction (H'5770) for the emulator and executes the user program. Accordingly, this command cannot be used for ROM. For ROM, transfer the program to emulation memory using the MOVE_TO_RAM command.

  Usable areas of STEP_OVER command: Internal ROM, internal RAM,
  area allocated with emulation memory, and
  RAM in user system

  — 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=00001000  CCR=80:I*******  EXR=07:*****210
     MACH=00000000  MACL=00000000
     ER0–ER3  00000000   00000001   00000002   00000003
     ER4–ER7  00000008   00000009   0000000A   0000000B
(b)  <address> <instruction mnemonic>
(c)  MEMORY
       <memory contents>
(d)  <cause of termination>
```

(a) Register information

Note: When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist, and therefore, cannot be displayed. However, instruction mnemonic display is performed including the MACH and MACL registers.

(b) Address and mnemonic of the instruction that was executed

(c) Memory contents display

(d) Cause of termination

— 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_CONDITION1,2,3,4,5,6 specifications) are invalid during STEP_OVER command execution.

— Interrupts are not accepted during STEP_OVER command execution, unless the I option is specified.

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	An invalid instruction (H'5740 to H'577F) was executed.
RESET IN BY E7000	A problem occurred in the user system. The emulator had input the RES signal and forcibly terminates execution.
SUBROUTINE END ADDRESS IS NOT RAM	Being in the ROM area, the end address cannot be executed. (Refer to descriptions.)
SUBROUTINE END ADDRESS IS BREAK INSTRUCTION	The instruction following a JSR, BSR, or TRAPA is an illegal instruction (H'5770)

Note

This command must be used only when the program execution returns from a subroutine called by a BSR, JSR, or TRAPA instruction to an instruction immediately following the BSR, JSR, or TRAPA instruction. However, because this command writes over the instruction following the JSR, BSR, or TRAPA instruction, do not use it if the instruction that will be modified is being used as data.

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  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000002 00000003
 ER4 - ER7  00000008 00000009 0000000A 0000FF00
 00001000                 MOV.W     R0,R1
 +++:ONE STEP END
:(RET)
 PC=00001004  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000002 00000003
 ER4 - ER7  00000001 00000009 0000000A 0000FF00
 00001002                 MOV.W     #0001:16,R4
 +++:ONE STEP END
:(RET)
 PC=00001008  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000044 000000FF
 ER4 - ER7  00000008 00020000 0000000A 0000FF00
 00001004                 JSR       @002010:24....(The subroutine is not
 +++:SUBROUTINE END                                  displayed)
:(RET)
 PC=0000100A  CCR=80:I*******  EXR=07:*****210
 MACH=00000000  MACL=00000000
 ER0 - ER3  00000000 00000000 00000044 000000FF
 ER4 - ER7  00000008 00020000 0000000A 0000FF00
 00001008                 NOP
 +++:ONE STEP END
:
```

7.2.45	**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 higher than H'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_CONDITION1,2,3,4,5,6
  BREAK_SEQUENCE
  TRACE_CONDITION
  LED1,2,3,4
  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: xyz)
                int  a;
                int  b;
             }  xyz  ;            (Variable: xyz)
```

Examples

1. To define symbol SUB00:

```
:SY !SUB00=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    H8300POD        LABEL         00000330
:
```

7.2.46	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.
When this option is omitted, instruction mnemonic information is displayed for branch instruction trace mode.

Description

- Display

— Displays trace information acquired when accessing an external bus 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 the following 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.

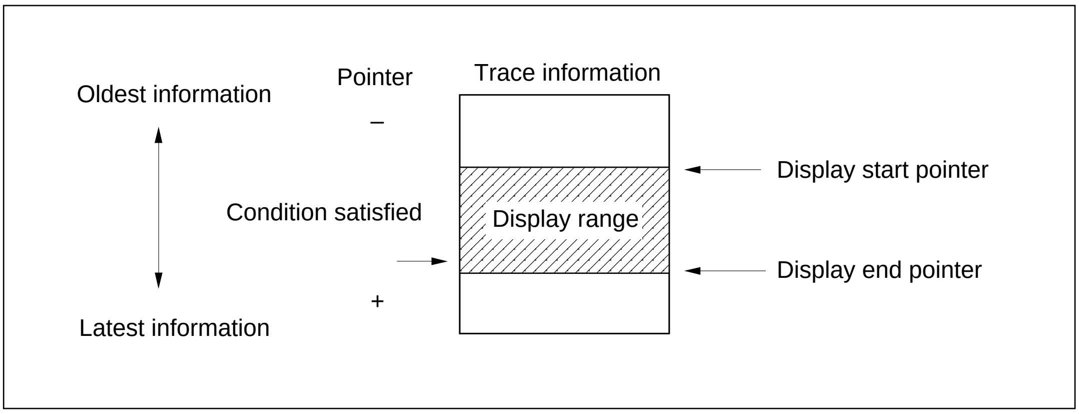


Figure 7-2 Display Range Specified by Instruction Pointers

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    IRQ     NMI   RA  PROB    CLK        TM
[-]D'xxxxx  xxxxxxxx  xxxxxxxx   xxx    x   xxx xxxxxxxx   x    xx xxxxxxxx  xx   xxxxxxxx=xxxx
    (a)        (b)        (c)     (d)   (e)  (f)   (g)     (h)  (i)   (j)    (k)       (l)
   ---------------------------------------------------------------------------------------------
                                                   TOTAL CLOCK NUMBER   =   xxxxxx
                                                                             (m)
```

(a) Bus cycle pointer
Number of bus cycles from 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. 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.

(d) Memory area type

Table 7-23 MA Display

Display	Description
ROM	Internal ROM area access
RAM	Internal RAM area access
IO	Internal I/O area access
EXT	External memory area access (including unusable 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
REF	Refresh cycle
DAT	Memory or I/O access cycle
DMA	DMA cycle
DTC	DTC cycle

(g) IRQn signal level

```
        IRQ
x7  x6  x5  x4  x3  x2  x1  x0

xn:  IRQn signal status        xn 0:  Low level
                                   1:  High level
```

Note: The IRQ pins that do not exist in the MCU are displayed as 1s.

(h) NMI signal level (0 = low level, 1 = high level)

(i) BREQ and BACK signal levels

```
RA
x1  x2

x1:  BACK signal status            xn 0: Low level
x2:  BREQ signal status                1: High level
```

(j) External probe signal levels

```
PROB
x7  x6  x5  x4  x3  x 2  x1  x0

xn:  PROB n signal status          xn 0: Low level
                                       1: High level
```

(k) The number of clock cycles required from the end of the previous bus cycle to the end of this bus cycle. Up to 255 clocks are counted. If the number is more than 256, it is displayed as **. Refer to section 1.5.1, Trace Timing.

(l) Address specified by the TRACE_MEMORY command and its memory contents

(m) The total number of clock cycles (value displayed at (k) 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 256 clock cycles) appears at (k), ***** is displayed as the total value.

— If the (CTRL) + P keys are pressed during trace information display, the emulator backs up 32 lines, displays 32 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 32 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 user program execution. Note that this emulator cycle does not affect user program execution cycles.

   *** E7000 ***

   This emulator cycle is displayed even when user program execution halts as a result of multiple passes specified with the BREAK command or pass points specified with the BREAK_SEQUENCE command being satisfied.

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. When H8S/2000 is selected as the MCU with the MODE command, the MACH and MACL registers do not exist. However, instruction mnemonic display is performed including the MACH and MACL registers with this command.

Examples

1. To display all trace information with only instruction mnemonics:

```
:T (RET)
   IP         ADDR     LABEL     MNEMONIC   OPERAND
 *-D'00007  00000000   !main     BSR        000004:8
 *-D'00006  00000004             JSR        @00008E:24
 *-D'00005  0000008E             PUSH.L     ER5
 *-D'00004  00000092             PUSH.L     ER4
 *-D'00003  00000096             PUSH.L     ER3
 *-D'00002  0000009A             PUSH.L     ER2
 *-D'00001  0000009E             MOV.L      @(0010:16,ER7),ER2
 * D'00000  000000A4             PUSH.L     ER2
:
```

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     IRQ    NMI RA   PROB  CLK
*         00000050             BLS       00004:8
 -D'00005 00000050 43F2 EXT  R  PRG  11111111  1  11 11111111 06
 -D'00004 00000052 0B56 EXT  R  PRG  11111111  1  11 11111111 06
*         00000044             SUB.B     R3L,R3L
 -D'00003 00000044 18BB EXT  R  PRG  11111111  1  11 11111111 06
*         00000046             MOV.B     R3L,@ER5
 -D'00002 00000046 68DB EXT  R  PRG  11111111  1  11 11111111 06
*         00000048             MOV.W     R6,R3
 -D'00001 00000048 0D63 EXT  R  PRG  11111111  1  11 11111111 06
  D'00000 200019AA 00   EXT  W  DAT  11111111  1  11 11111111 06
*         0000004A             EXTS.L    ER3
  D'00001 0000004A 17F3 EXT  R  PRG  11111111  1  11 11111111 06
*         0000004C             ADD.L     ER3,ER5
  D'00002 0000004C 0AB5 EXT  R  PRG  11111111  1  11 11111111 06
*         0000004E             CMP.L     ER4,ER5
  D'00003 0000004E 1FC5 EXT  R  PRG  11111111  1  11 11111111 06
  D'00004 00000050 43F2 EXT  R  PRG  11111111  1  11 11111111 06
  D'00005 *** E7000 ***
-----------------------------------------------------------------------
                                   TOTAL CLOCK NUMBER = 00003C
:
```

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     IRQ    NMI RA   PROB  CLK
*         0000004A             EXTS.L    ER3
 -D'00020 0000004A 17F3 EXT  R  PRG  11111111  1  11 11111111 06
*         0000004C             ADD.L     ER3,ER5
 -D'00019 0000004C 0AB5 EXT  R  PRG  11111111  1  11 11111111 06
*         0000004E             CMP.L     ER4,ER5
 -D'00018 0000004E 1FC5 EXT  R  PRG  11111111  1  11 11111111 06
*         00000050             BLS       000044:8
 -D'00017 00000050 43F2 EXT  R  PRG  11111111  1  11 11111111 06
 -D'00016 00000052 0B56 EXT  R  PRG  11111111  1  11 11111111 06
-----------------------------------------------------------------------
                                   TOTAL CLOCK NUMBER = 00001E
:
```

TRACE_CONDITION		
7.2.47	**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 7-26 (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.

TRACE_CONDITION

Table 7-26 Specifiable Conditions (TRACE_CONDITION)

Item and Input Format	**Description**	**Specifiable Trace Mode**
Address condition • Range trace	The condition is satisfied when the address bus value is in the range from <address 1> to <address 2>.	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 is 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>
WD=<2-byte value>
LD=<1-byte value>
(Data on data bus D7 to D0)
HD=<1-byte value>
(Data on data bus D15 to D8)*
	The condition is satisfied when the data bus value matches the specified value.*
	D, LD, and HD are valid when byte access is performed, while WD is valid when word access is performed.
		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 DTC: DTC 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 no bus-cycle type.	Trace stop or trigger trace

Note: Refer to the description on the following page.

Table 7-26 Specifiable Conditions (TRACE_CONDITION) (cont)

<table>
<tr><th>Item and Input Format</th><th>Description</th><th>Specifiable Trace Mode</th></tr>
<tr><td>External probe condition
PROB=<value>
&lt;value>: Values for probes 1 to 8</td><td>The condition is satisfied when all the emulator's external probe signals match the specified values. Specify <value> as one byte of data. Each bit corresponds to a probe number, as follows:
7 6 5 4 3 2 1 0 ← Bit position
x x x x x x x x ← Specified value
| | | | | | | |
8 7 6 5 4 3 2 1 ← Probe number
x: 0=Low level
1=High level
Example: To generate a break when probes 1 and 6 are high and the others are low, specify:
PROB=H'21</td><td>Trace stop or trigger trace</td></tr>
<tr><td>System control signal condition
BREQ, BACK</td><td>The condition is satisfied when the BREQ or BACK signal is low.</td><td>Trace stop or trigger trace</td></tr>
<tr><td>Delay count specification
DELAY=<value>
<value>: H'1 to H'7FFF</td><td>This condition can be specified in combination with any other conditions listed in this table. The complete condition combination is satisfied when the specified number of bus cycles has been executed after the other specified condition is satisfied.</td><td>Trace stop or trigger trace</td></tr>
</table>

— 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.

Address and data conditions for trigger trace and trace stop modes are described below.

a. Address condition for range trace mode
Address A0 is ignored. Though even addresses are always specified, a condition can also be satisfied at odd addresses. Data condition cannot be specified.
Example: H'1000 is specified as an address condition: The condition is satisfied when address H'1000 is accessed in word or byte units or when address H'1001 is accessed in byte units.

b. Address and data conditions for trigger trace and trace stop modes

- Word access

  Word data is accessed in one bus cycle. Only WD (word data) can be specified as a data condition.

- Byte access

  All addresses can be accessed by a byte access. Both even and odd addresses can be specified as an address condition. Note, however, that only byte data (D) is valid for data condition.

Notes: 1. In the actual MCU, data bus lines D15 to D8 are used when an 8-bit access area is accessed. However, in the emulator, D15 to D8 are used when an even address is accessed, and D7 to D0 are used when an odd address is accessed. When specifying data in byte access cycles, follow note 2 below regardless of whether to access a 16-bit bus area or an 8-bit bus area.

2. D, HD, or LD can be specified as byte data. Use these three data types depending on the trace conditions to be specified.

D: If the specified address is even, data on data bus D15 to D8 is specified. If the specified address is odd, data on data bus D7 to D0 is specified. If no address is specified or if an address range or mask is specified, data on data bus D15 to D8 is automatically specified as shown below.

Example 1: TC A=101 D=10
A condition is satisfied when byte data 10 is written to or read from address 101.

Example 2: TC A=100 : 1FF D=20
A condition is satisfied when byte data 20 is written to or read from an even address within the range from H'100 to H'1FF.

HD: Byte data (data bus D15–D8) access on an even address is always specified. Data access on an odd address is ignored.

Example: TC A=1000 : 10FF HD=80
A condition is satisfied if byte data 80 is written to or read from an even address within the range from H'1000 to H'10FF.

LD: Byte data (data bus D7–D0) access on an odd address is always specified. Data access on an even address is ignored.

Example: TC A=1000 : 10FF LD=80
A condition is satisfied when byte data H'80 is written to or read from an odd address within the range from H'1000 to H'10FF.

Note that D, HD, or LD cannot be specified in word access.

3. When an 8-bit access area is accessed in word units, two byte access cycles are executed in the actual MCU. However, in the emulator, it is executed in one bus cycle. In this case, byte data (D, HD, and LD) conditions are not satisfied. Specify word data (WD) conditions.

— A bit mask can be specified for address or data 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 an asterisk (*). The number of masked bits depends on the input radix. Examples of masks are given below. Table 7-27 shows mask specifications.

Example 1: A condition is satisfied when the D0 bit is 0 in a byte data condition.

: TRACE_CONDITION D = B'*******0 (RET)

Example 2: A condition is satisfied when probe 3 is 0 in a probe condition.

: TRACE_CONDITION PROB = B'*******0 (RET)

Table 7-27 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, HD), PROB
Hexa-decimal	4 bits	H'F**50	D15 to D8 bits are masked	Address, data (D, WD, LD, HD), PROB

— In parallel mode, this command is executed as follows:

Parallel mode is entered by the (RET) key, or the trace acquisition 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
    A condition is modified with the TRACE_MEMORY command

  As soon as the above command is terminated, trace information acquisition starts.
  In this case, conditions that have been satisfied are 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)
  ```

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 1 is pulled low (trace stop):

   ```
   :TC  PROB=B'*******0 ; S (RET)
   :GO
    ** 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 trace):

   ```
   :TC A=FF00 D=80 W ; T (RET)
   :
   ```

TRACE_MEMORY

7.2.48	**TRACE_MEMORY** **TM**	**Specifies, displays, and cancels memory address to be traced**

Command Format

- Setting : TRACE_MEMORY <address>[Δ<access type>][;<size>] (RET)
- Display : TRACE_MEMORY (RET)
- Cancellation : TRACE_MEMORY [Δ]– (RET)

<address> : Memory address to be traced into trace buffer
<access type> : Bus access type to be traced into trace buffer
DMA : DMA cycle
DTC : DTC cycle
Default: Instruction execution, DMA, and DTC cycles
<size> : Traced size
B : Byte
W : Word (equal to two bytes; valid when <address> has an even value)
Default : Word

Description

- Setting

  — Acquires the contents of the specified memory address in bus-cycle units and displays them during user program execution.

  — Use the TRACE or TRACE_SEARCH command to display or search for address contents stored in the trace buffer. Specify the TM option of the GO command to display address contents during user program execution.

- Display

  Displays the specified address, access type, and size as shown below.

```
: TRACE_MEMORY (RET)
  ADDRESS = xxxxxxxx  yyy ; z  !ssssss
```

xxxxxxxx: Specified address
yyy: Access type
DAT: Instruction execution cycle
DMA: DMA cycle
DTC: DTC cycle
z: Traced size
!ssssss: Symbol name (displayed only when an address is specified by a symbol)

- Cancellation

  Cancels the specified memory address.

Note

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, a DMA cycle, or in a DTC 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.
- 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.

Examples

1. To trace the memory contents in address H'FF0E and display the results during user program execution:

```
:TM  FF0E (RET)
:G ;TM (RET)
 ** PC = 00001204 0000FF0E = 8044
```

2. To display the specified address:

```
:TM (RET)
 ADDRESS = 0000FF0E   DAT ; W      !table/stack_top
:
```

3. To cancel the specification:

```
:TM - (RET)
:
```

7.2.49	TRACE_MODE TMO	Specifies and displays trace information acquisition mode

Command Format

- Setting : TRACE_MODE [ΔREF=<REF option>] [ΔPTR=[-]<start pointer>[:[–]<end pointer>]][;C] (RET)
- Display : TRACE_MODE (RET)

<REF option>: Display of refresh cycle trace information
E: Enables display of refresh cycle trace information
D: Disables display of refresh cycle trace information (Default at emulator shipment)
<start pointer>: Start pointer during trace information display and search (Default at emulator shipment is –D'4095)
<end pointer>: End pointer during trace information display and search (Default at emulator shipment is D'4095)
C: Saves the specified mode in a configuration file

Description

- Setting

— Enables or disables trace information display for refresh cycles.

- To disable trace information display for refresh cycles:

: TRACE_MODE REF = D (RET)

- To enable trace information display for refresh cycles:

: TRACE_MODE REF = E (RET)

— 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 –32767 to 32767. When exceeding this range, start and end pointers are automatically specified as –32767 and 32767, respectively.

: TRACE_MODE PTR = –D'2048:D'2048 (RET)

— When the C option is specified, writes the specifications in a configuration file on the system disk. Hereafter, when the emulator is activated with the system disk, the saved specifications go into effect.

When the E7000PC is used, the specifications are written in the configuration file in the directory where the current system program is stored on the IBM PC.

The following message is output to confirm with the user whether to rewrite the existing configuration file contents on the IBM PC.

```
: 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.

- Display

Displays the specified trace mode as shown below.

```
: TRACE_MODE (RET)
 REF = x1   PTR = –D'yyyyy  :  D'yyyyy
```

x1: E: Enables trace information display for refresh cycles
D: Disables trace information display for refresh cycles

yyyyy: Default values of start and end bus cycle pointers while trace information is displayed or searched

Examples

1. To trace information on internal I/O access bus 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 REF=D PTR = -D'10:D'10;C (RET)
 CONFIGURATION WRITE OK (Y/N)? Y (RET)
:
```

2. To display the specified contents:

```
:TMO (RET)
 REF=D PTR = –D'00010:D'00010
:
```

7.2.50	**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)

  — If the L option is specified, displays only the last bus cycle information to be searched for.

  — Items listed in table 7-28 can be specified for <condition>, and they can be combined by ANDing them.

Table 7-28 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=<1-byte value>
(Data on data bus D7 to D0)
HD=<1-byte value>
(Data on data bus D15 to D8)
	Searches for a bus cycle where the data bus value matches the specified value.
	D, LD, and HD are valid when byte access is performed, while WD is valid when word access is performed*.

Memory area condition ROM: Internal ROM area RAM: Internal RAM area IO: Internal I/O area EXT: External memory area (Unusable area)	Searches for a bus cycle where the specified memory area is accessed.
Read/write condition R: Read W: Write	Searches for a read cycle (R is specified) or a write cycle (W is specified).
Access type PRG: Program fetch cycle DAT: Execution cycle DMA: DMA cycle DTC: DTC 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 no bus-cycle type.

Note: Refer to the description on the following page.

Table 7-28 Specifiable Conditions (TRACE_SEARCH) (cont)

Item and Input Format	Description
External probe condition PROB=<value>* <value>: Values for probes 1 to 8	Searches for a bus cycle in which all emulator external probe signals match the specified value. Specify <value> as one byte of data. Each bit of <value> corresponds to a probe number, as follows: 7 6 5 4 3 2 1 0 ← Bit position x x x x x x x x ← Specified value │ │ │ │ │ │ │ │ 8 7 6 5 4 3 2 1 ← Probe number x: 0=Low level 1=High level Example: To search for a bus cycle in which probes 1 and 6 are high and others are low, specify: PROB=H'21
External interrupt condition NMI: {H, L}	• 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.
IRQ=<value>	• Searches for a bus cycle in which all IRQ signals are at the specified levels. Specify <value> as 1-byte data (H'0–H'0F) as follows. Each bit corresponds to a signal (IRQ0 to IRQ7). 7 6 5 4 3 2 1 0 ← Bit position x x x x x x x x ← Specified value │ │ │ │ │ │ │ │ 7 6 5 4 3 2 1 0 ← IRQ number x: 0=Low level 1=High level Example: To search for a bus cycle in which IRQ1 and IRQ5 are high and others are low, specify: IRQ=H'22
Data condition for the address specified with TRACE_MEMORY command TM= {<data value>, CH}	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.

Note: Refer to the description on the following page.

— When an address or data condition is specified, the emulator searches for a bus cycle where address bus value and data bus values match the specified values, respectively. Note the following when specifying search conditions.

- Word access

  Word data is accessed in one bus cycle. Only WD (word data) can be specified as a data condition.

- Byte access

  All addresses can be accessed by a byte access. Both even and odd addresses can be specified as an address condition. Note, however, that only byte data (D, HD, LD) is valid for data condition.

Notes: 1. D, HD, or LD can be specified as byte data. Use these three data types depending on the search conditions to be specified.

D: Byte data on an even or odd address is specified. Word access is ignored.
Example 1: TS A=101 D=10
Searches for a bus cycle in which byte data 10 is written to or read from address 101.
Example 2: TS A=100 : 1FF D=20
Searches for a bus cycle in which byte data 20 is written to or read from addresses 100 to 1FF.

HD: Byte data (data bus D15–D8) access on an even address is always specified. Data access on an odd address is ignored.
Example: TS A=1000 : 10FF HD=80
Searches for a bus cycle in which byte data 80 is written to or read from an even address within the range from 1000 to 10FF.

LD: Byte data (data bus D7–D0) access on an odd address is always specified. Data access on an even address is ignored.
Example: TS A=1000 : 10FF LD=80
Searches for a bus cycle in which byte data 80 is written to or read from an odd address within the range from 1000 to 10FF.

Note that D, HD, or LD cannot be specified in word access.

2. **When an 8-bit access area is accessed in word units, two byte access cycles are executed in the actual MCU. However, in the emulator, it is executed in one bus cycle. In this case, byte data (D, HD, and LD) conditions are not satisfied. Specify word data (WD) conditions.**

— A bit mask can be specified for data, IRQ, or external probe condition specification in single-bit or 4-bit units. 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 (*). Examples of masks are given below. Table 7-29 shows the mask specifications.

Example 1: To search for a bus cycle where D0 bit is 0 in byte data condition.

```
: TRACE_SEARCH  D = B'*******0 (RET)
```

Example 2: To search for a bus cycle where IRQ2 is 0 in IRQ condition (IRQ pins other than IRQ2 are ignored)

```
: TRACE_SEARCH  IRQ = B'*****0** (RET)
```

Table 7-29 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 (D, WD, LD, HD), IRQ, PROB
Hexa-decimal	4 bits	H'F**50	D15 to D8 bits are masked	Address, data (D, WD, LD, HD), IRQ, PROB

Note: A mask cannot be specified for an address range.

— The display contents are the same as the bus-cycle display of the TRACE command. Refer to section 7.2.46, 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.

Examples

1. To search for bus cycles when data is written to addresses in the range from H'FF00 to H'FF0F:

```
:TS A=FF00:FF0F W  (RET)
     BP       AB      DB     MA   R/W  ST    IRQ   NMI  RA    PROB    CLK
 -D'00065  00FF01  43     EXT   W   DAT  111111  1   11  11111111  06
 -D'00044  00FF0E  0B     EXT   W   DAT  111111  1   11  11111111  06
 -D'00032  00FF02  80     EXT   W   DAT  111111  1   11  11111111  06
 -D'00020  00FF0B  00     EXT   W   DAT  111111  1   11  11111111  06
:
```

2. To search for the last bus cycle where IRQ0 is low:

```
:TS IRQ = B'*******0 ;L  (RET)
     BP      AB      DB     MA   R/W  ST    IRQ   NMI  RA    PROB    CLK
 -D'00020  003402  0000   EXT   R   PRG  111110  1   11  11111111  06
:
```

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.

Note: 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 a 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. For details, refer to section 8.4.11, FLOPPY_FORMAT.

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 computer are divided into records. The format of a record is shown below. One record can extend over several clusters.

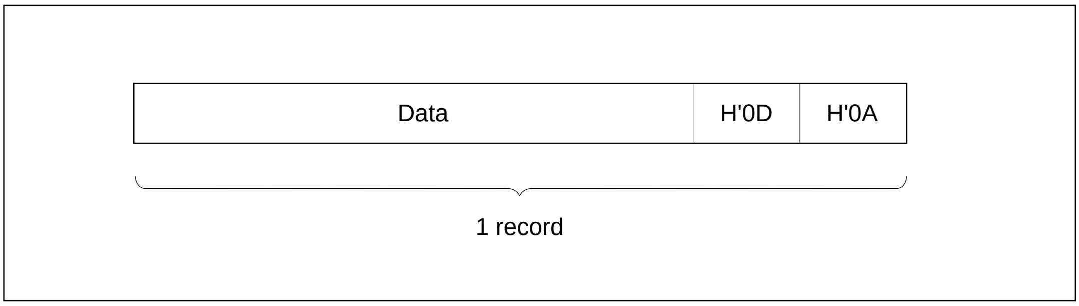
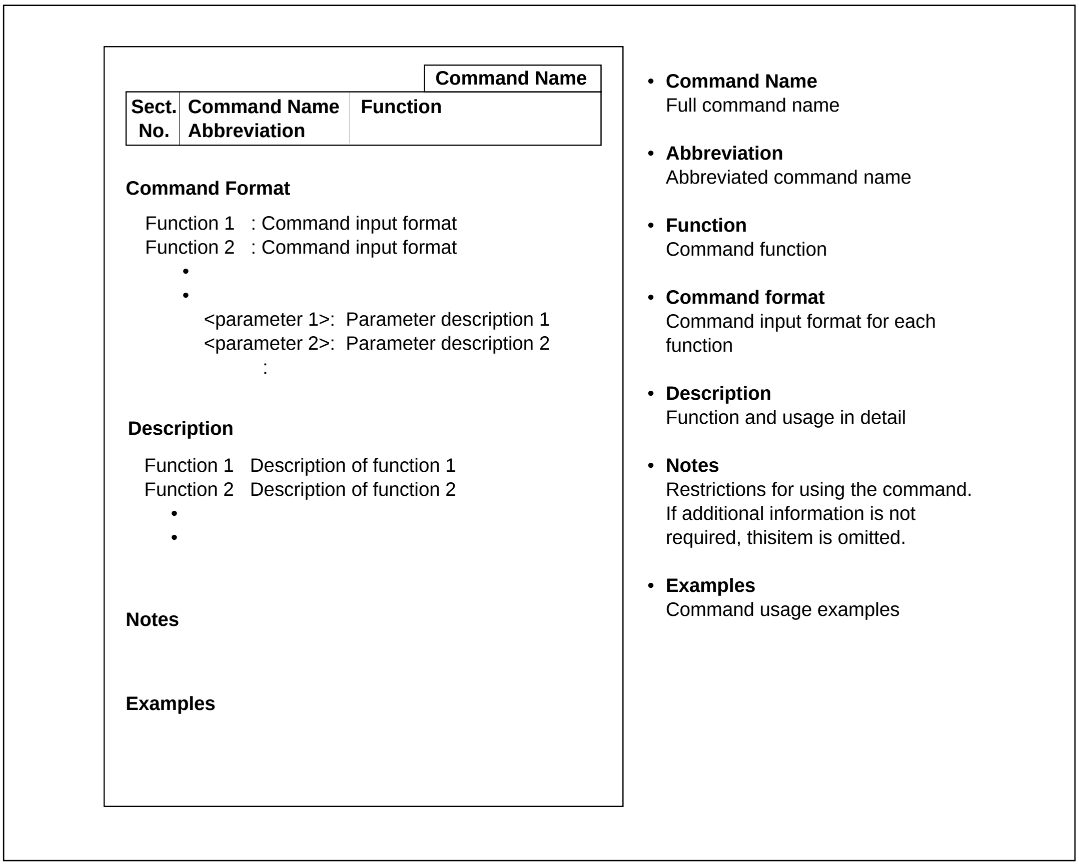


Figure 8-1 Record Format

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 computer.

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**
  Restrictions for using the command. If additional information is not required, thisitem 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

FILE_COPY

— 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'
:
```

FILE_DIRECTORY		
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	Function
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

- File 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

- File 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 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)
<load module type>:	Load module type to be loaded

S:	S-type load module
H:	HEX-type load module
B:	Binary-type load module
Default:	Binary-type load module

Description

- Load

— Loads load module file contents saved into a floppy disk 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 internal I/O 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)
 LOADING ADDRESS 00000000
 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 Short Formats for File Names

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)
:
```

FILE_SAVE		
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
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 internal I/O 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)
 SAVING ADDRESS 00000400
 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 40FF;S (RET)
 SAVING ADDRESS 00004000
 TOP ADDRESS = 00004000
 END ADDRESS = 000040FF
:
```

FILE_TYPE

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. For details, refer to 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 (whose contents are output with the PRINT command) on the console:

```
:FTY RESULT (RET)
:MAP 0 7FFFF ;S
 REMAINS EMULATION MEMORY S=D'30000KB
:F 0 FFFF
:FL TEST
 TOP ADDRESS = 00000000
 END ADDRESS = 00001FFF
: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)
  <load module type>: Load module type to be verified
  S: S-type load module
  H: HEX-type load module
  B: Binary-type load module
  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:

```
VERIFYING 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>
```

FILE_VERIFY

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 internal I/O area cannot be verified.

Examples

1. To verify data in file PRG.B against memory contents:

```
:FV PRG.B (RET)
 VERIFYING ADDRESS 00000000
 TOP ADDRESS = 00000000
 END ADDRESS = 00001FFF
:
```

2. To verify data against that in S-type file TST.S (two verification errors occurred):

```
:FV TST.S (RET)
 VERIFYING ADDRESS 00000000
 <ADDR>    <FILE>   <MEM>
 00002024   30'0'    31'1'
 00003020   00'.'    01'.'
 TOP ADDRESS = 00002000
 END ADDRESS = 00003FFF
:
```

FLOPPY_CHECK

8.4.10	**FLOPPY_CHECK** **FCH**	**Displays floppy disk information**

Command Format

- Disk information display :FLOPPY_CHECK (RET)

Description

- Disk information 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 disk 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 name> 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 with the volume name 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. The volume name is WORK:

```
:FF (RET)
 VOLUME LABEL                  FORMAT OK (Y/N) ? N (RET)
 KEY IN VOLUME LABEL (11 CHARACTERS) ? WORK (RET)
:
```

Section 9 Data Transfer from Host Computer Connected by RS-232C Interface

9.1 Overview

When the E7000 is connected to a host computer by the RS-232C interface, data can be transferred between the host computer and the E7000 or between the host computer and user system memory connected to the E7000. This enables the following transmission of host computer load module files:

- Loads a load module file in the host computer to user system memory
- Saves data in the user system memory as a load module file in the host computer
- Transfers host computer text files to E7000 floppy disk files
- Transfers files between the E7000 and host computer

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 Computer

Command	Function	Usable/Unusable in Parallel Mode
HOST	Specifies and/or displays host computer interface parameters.	Unusable
LOAD	Loads program from host computer. — Transparent mode and local mode	Unusable
SAVE	Saves program in host computer. — Transparent mode and local mode	Unusable
TERMINAL	Transfers to terminal mode. — Transparent mode	Unusable
TRANSFER	Transfers file to and from host computer. — Transparent mode and local mode	Unusable
VERIFY	Verifies memory contents against host computer file. — Transparent mode and local mode	Unusable
INTFC_LOAD	Loads program from host computer. — Remote mode	Unusable
INTFC_SAVE	Saves program in host computer. — Remote mode	Unusable
INTFC_ TRANSFER	Transfers file to and from host computer. — Remote mode	Unusable
INTFC_ VERIFY	Verifies memory contents against host computer file. — Remote mode	Unusable

Note: Transparent, local, and remote modes are described in detail in section 9.2, Host Computer Interface Modes and Operating Procedures.

9.2 Host Computer Interface Modes and Operating Procedures

The E7000 interfaces with the host computer in transparent mode, local mode, or remote mode, when the RS-232C interface is used.

9.2.1 Transparent Mode

This mode is valid for a host computer which can be connected to the console by an RS-232C interface cable and can be disconnected. A single console is shared between the E7000 and the host computer in transparent mode.

The configuration and data transfer in this mode are shown in figure 9-1. To enable data transfer, the host computer command must be entered in addition to the emulator command.

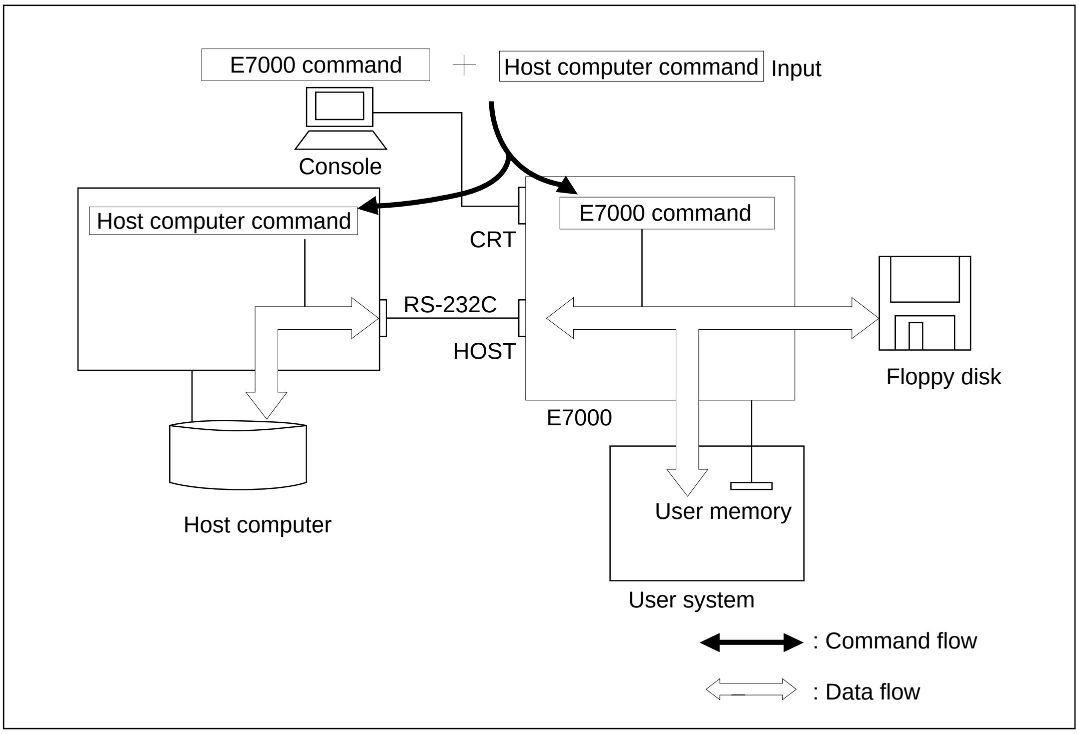


Figure 9-1 Configuration and Data Transfer in Transparent Mode

Procedure:

(a) Start up the E7000 system program.

(b) Set up the host computer'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 computer.

(d) Set up host computer.	Start up the host computer and put it in command input wait 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 computer command to the E7000 data transfer command (LOAD, SAVE, VERIFY, or TRANSFER) Example: LOAD : TYPE X.MOT (RET)

9.2.2 Local Mode

In local mode, both E7000's console and host computer's console are used for data transfer. An EPROM programmer can be connected in local mode. Emulator commands and host computer commands are both executed at data transfer. Execute the data receiving command first. The configuration and data transfer in this mode are shown in figure 9-2.

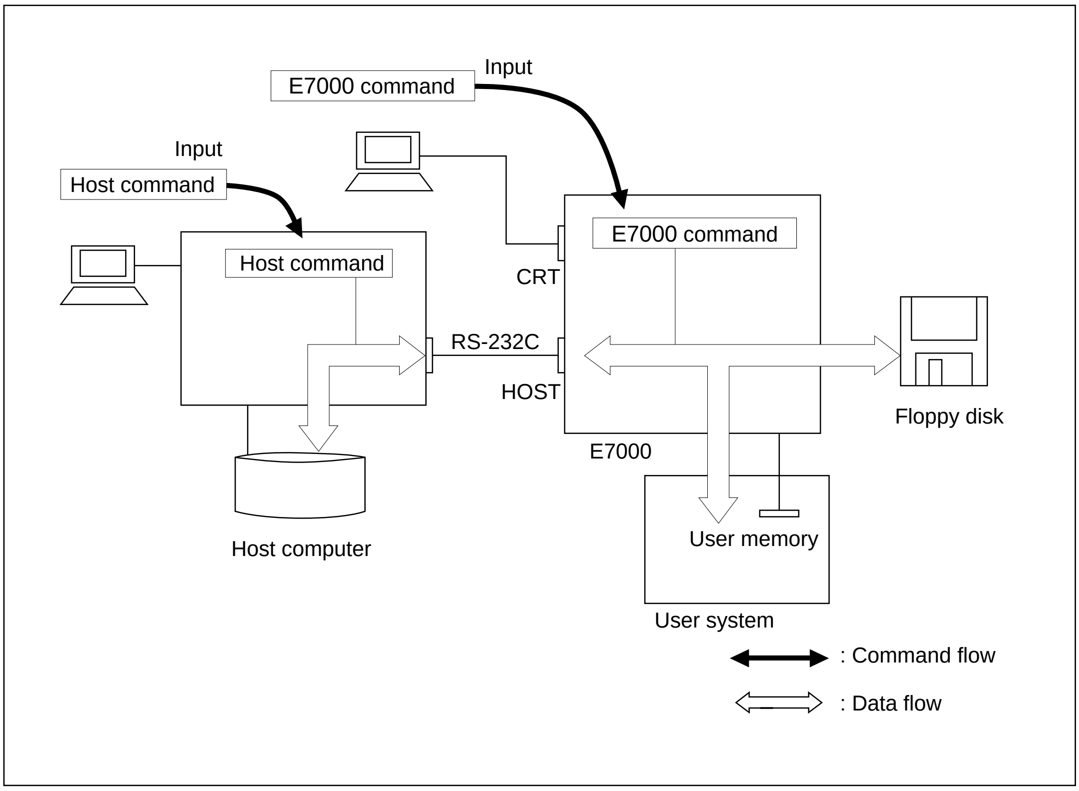


Figure 9-2 Configuration and Data Transfer in Local Mode

Procedure:

Host Computer	**E7000**	**Description**
Start up host computer.	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 computer to E7000:		
	Execute data receive command.	Execute the E7000 command (LOAD, VERIFY, or TRANSFER). The E7000 can receive data from the host computer. Be sure to execute the receive command first.
Execute data transmission command.		Execute the host computer command to output data to the E7000 connection port.
Data transfer from E7000 to host computer:		
Execute data receive command.		Execute the host computer 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 computer that has its own console, such as a personal computer, is connected to the E7000 as the host computer. This mode requires H-series interface software that can be separately purchased. In this mode, a host computer'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. Be sure to use the INTFC_LOAD, INTFC_SAVE, INTFC_VERIFY, and INTFC_TRANSFER commands in remote mode. The configuration and data transfer in this mode are shown in figure 9-3.

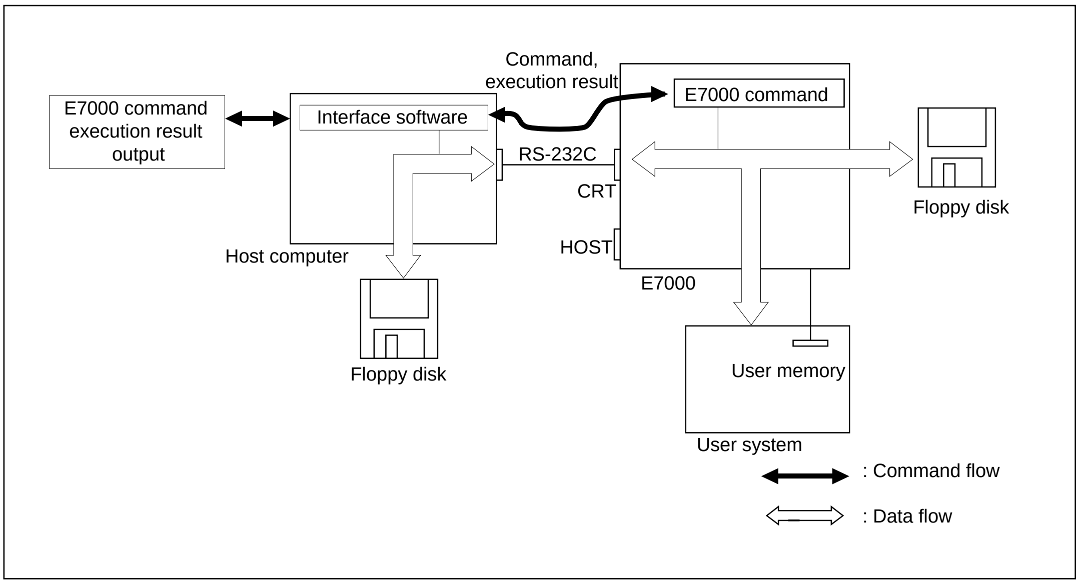


Figure 9-3   Configuration and Data Transfer in Remote Mode

Procedure:

Start up host computer.

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 computer. E7000 commands can now be entered from the host computer.

Data transfer from host computer 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 computer to the E7000.
Example:
INTFC_LOAD:<host computer file name> (RET)

Data transfer from E7000 to host computer:
Execute E7000 data transmission command.

The E7000 data transmission command (INTFC_SAVE or INTFC_TRANSFER) can transfer data from the E7000 to the host computer.
Example:
INTFC_SAVE 0 1FFF:<host computer file name> (RET)

9.3 Data Transfer Control

9.3.1 Control Methods

The E7000 provides an RS-232C interface for the host computer. 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.

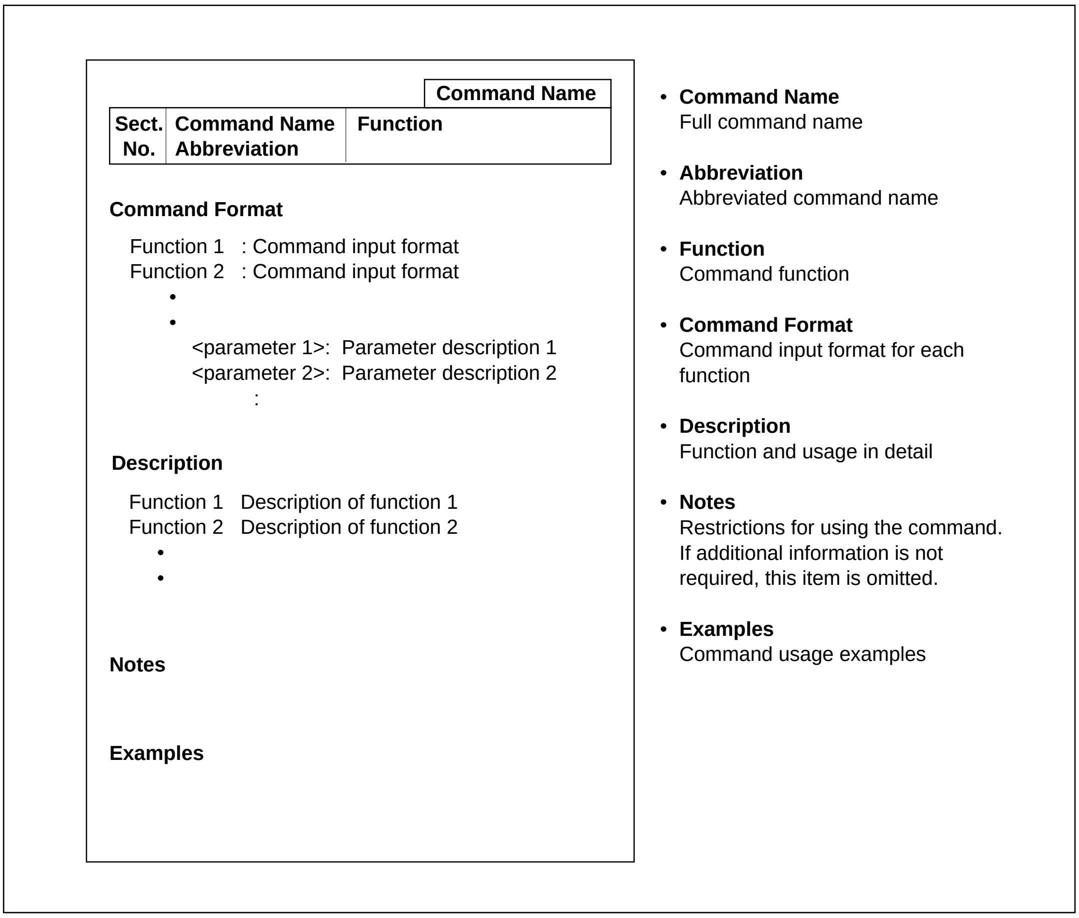
- RTS/CTS control
  Stops data transfer when the data-receiving system outputs a low-level RTS signal, and restarts data transfer when the data-receiving system outputs a high-level RTS signal.

9.3.2 Timeouts

The E7000 monitors for timeouts as it receives data from the host computer. After command execution, the E7000 waits an unlimited time for the first data byte. However, once it has received the first byte, it will become timeout after waiting three seconds for the next data byte, and command execution will terminate.

9.4 Host-Computer Related Commands

This section provides details of host-computer 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**
  Restrictions for using the command. If additional information is not required, this item is omitted.
- **Examples**
  Command usage examples

Figure 9-4 Format of Host-Computer 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 computer interface 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.

HOST

Description

- Specification

  Specifies the host computer’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 computer 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) Displays current communication parameters.
(b) Specify baud rate.
(c) Specify character length.
(d) Specify parity.
(e) Specify stop bit.
(f) Specify control method.
(g) Displays selected communication parameters.

Note

At system initiation, the host computer 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
:
```

LOAD

9.4.2	**LOAD** **L**	**Loads program from host computer** **— Transparent mode and local mode**

Command Format

- Load :LOAD[Δ<offset>][;[<load module type>][ΔN][ΔWA]]
  [:<command transferred to host computer>] (RET)

<offset>: Value to be added to the load module address (can only be specified for an S-type or HEX-type load module)

<load module type>: Load module type

R: SYSROF-type load module
S: S-type load module
H: HEX-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.

WA: Waits for the LF code. Refer to Description below.

<command transferred to host computer>: Specifies a command to be transferred to the host computer (only in transparent mode).

Description

- Load

— Loads a user program into user memory from the host computer. Loading in different host computer interface modes is described below.

Transparent Mode: After the command below is transferred to the host computer, the user program from the host computer is loaded into memory.

:LOAD;<load module type>:<command transferred to host computer> (RET)

<command transferred to host computer>: This command is transferred to the host computer. The characters following the colon (:) are sent directly to the host computer. The command to output source file contents to the terminal is specified.

The E7000 transfers <command transferred to host computer> and (RET) (CR code: H'0D) to the host computer. At the same time, when echo back display is specified, it displays the echo back from the host computer on the console. When option WA is not specified, load module transfer starts 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 computer. As soon as the LF code is received, load module transfer starts. 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 computer to no echo and transfer the load module without option WA.

Local Mode: The E7000 does not issue data output requests to the host computer. Therefore, after the LOAD command is entered, set the host computer 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 computer>] (RET)
```

If an offset is specified, a load address is calculated as follows:

Load address = <load module address> + <offset>

LOAD

— 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 computer>] (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, the confirmation request messages (b) are not displayed. If N is entered, the confirmation request messages are displayed. Enter the unit names of symbols to be defined.

(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.

Notes

1. The load module cannot be loaded to the internal I/O area.

2. Verification is not performed during load. 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, 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 computer 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)
 LOADING ADDRESS 00007000
 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
                                           computer transfers data to the E7000.
 LOADING ADDRESS 00000000
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

9.4.3	SAVE SV	Saves program in host computer — 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 computer>] (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 code (H'0A) to the end of each record.

<command transferred to host computer>: Specifies a command to be transferred to the host computer (only in transparent mode).

Description

- Save

— Saves the specified memory contents in the host computer in the specified load module type. An S-type or HEX-type load module can be saved. An SYSROF-type load module cannot be saved. Data receive request to the host computer in different host computer interface modes is described below.

Transparent Mode: After the specified command is transferred to the host computer, the memory area contents of the specified load module type are saved in the host computer.

```
:SAVE <start address> <end address>[;<load module type>]
                                        :<command transferred to host computer> (RET)
```

<command transferred to host computer>: This command is transferred to the host computer. The characters following the colon (:) are sent directly to the host computer. 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 computer. Therefore, before the SAVE command is input, set the host computer 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 internal I/O area cannot be saved.

2. Verification is not performed. Verify the program with the VERIFY command if necessary.

SAVE

Examples

1. To save memory contents in the address range from H'7000 to H'7FFF in the host computer in S-type load module format (in transparent mode). COPY TT: F11.S is a host computer command:

```
:SV 7000 7FFF :COPY TT: F11.S (RET)
 SAVING ADDRESS 00007000
 TOP ADDRESS=00007000
 END ADDRESS=00007FFF
:
```

2. To save memory contents in the address range from H'0000 to H'1E00 in the host computer in HEX-type load module format (in local mode):

```
                           ← Before entering the SAVE command, set the host
:SV 0 1E00 ;H (RET)          computer to be ready to receive data from the E7000.
 SAVING ADDRESS 00000000
 TOP ADDRESS=00000000
 END ADDRESS=00001E00
:
```

9.4.4	TERMINAL TL	Transfers to terminal mode — Transparent mode

Command Format

- Transfer to terminal mode :TERMINAL [Δ<end code>] (RET)

  <end code>: End key code for TERMINAL mode (1-byte data)
  Default is H'1A ((CTRL)+ Z).

Description

- Transfer to terminal mode

  Transfers characters entered from the keyboard to the host computer, and outputs data received from the host computer to the console. A console connected to the E7000 can be used as a host computer'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.

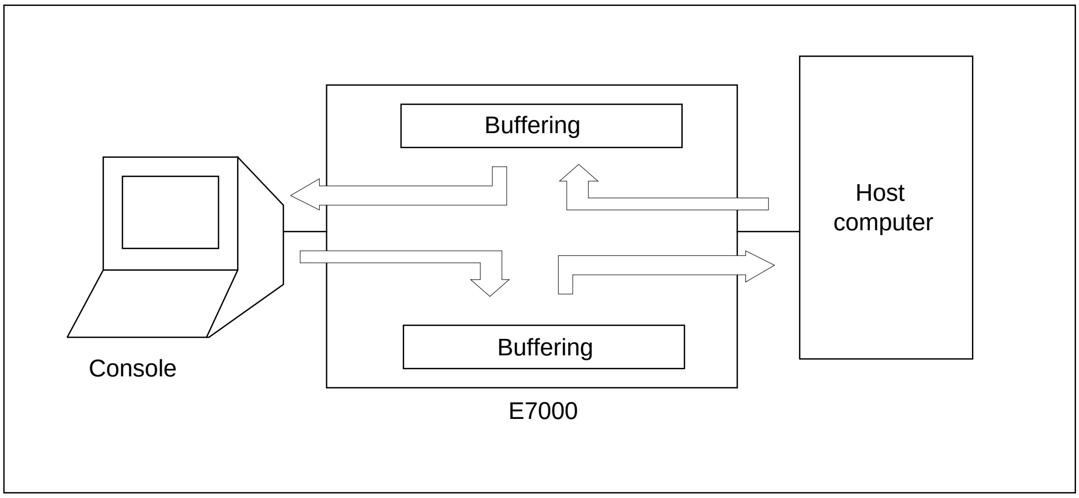


Figure 9-5 TERMINAL Command Processing

When changing the end 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 mode end code:

```
:TL 18 (RET)
$DIR              (Executes host computer command)
     :
$(CTRL) + X       (Terminates terminal mode)
:
```

9.4.5	TRANSFER TR	Transfers file to and from host computer — Transparent mode and local mode

Command Format

- Transfer :TRANSFER <file name>[;[(S/R)][ΔWA][ΔC]]
  [:<command transferred to host computer>] (RET)

<file name>: Name of file on floppy disk in the E7000

S: Transfer from the E7000 to the host computer

R: Reception from the host computer (default)

WA: Waits for the LF code (H'0A) after a command is transferred to the host computer. 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 computer and removes CR codes from files transferred to the host computer.

<command transferred to host computer>: Specifies a command to be transferred to the host computer (only in transparent mode).

Description

- Transmission

  — Only text files can be transferred from the E7000 to the host computer. Transfer processing in different host computer interface modes is described below.

  Transparent Mode: After the host computer command below is transferred, file contents are sent to the host computer.

  :TRANSFER <file name>;S :<command transferred to host computer> (RET)

  <command transferred to host computer>: This command is transferred to the host computer. The characters following the colon (:) are sent directly to the host computer. The command to store data from a terminal to a file is specified.

Local Mode: The E7000 does not issue data output requests to the host computer. Therefore, before this command is entered, set the host computer to input data from the E7000.

```
:TRANSFER <file name>;S (RET)
```

Remote Mode: Use the INTFC_TRANSFER command. For details, refer to section 9.4.9, INTFC_TRANSFER.

- Receive

— Transfers a file from the host computer to the E7000. Only text files can be transferred. Data transfer processing in different host computer interface modes is described below.

Transparent Mode: After the host computer command below is entered, transferred data is written to the E7000 file.

```
:TRANSFER <file name>;R :<command transferred to host computer> (RET)
```

<command transferred to host computer>: This command is transferred to the host computer. The characters following the colon (:) are sent directly to the host computer. The command to output the file contents to a terminal is specified.

The E7000 transfers <command transferred to host computer> and (RET) (CR code: H'0D) to the host computer. At the same time, when the echo back display is specified, it displays the echo back from the host computer on the console. When option WA is not specified, the E7000 starts to receive a load module within 50 ms after (RET) is received. When option WA is specified, the E7000 waits for the LF code (H'0A) sent from the host computer. As soon as the LF code is received, the E7000 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 computer to no echo and transfer the module without option WA.

Local Mode: The E7000 does not issue data output requests to the host computer. Therefore, after the TRANSFER command is entered, set the host computer to output data.

```
:TRANSFER <file name>;R (RET)
```

Remote Mode: Use the INTFC_TRANSFER command. For details, refer to section 9.4.9, INTFC_TRANSFER.

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 other types of data, an error is generated and the command is aborted.

This command is terminated with EOF (H'1A).

— With a UNIX-based host computer, 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 computer in transparent mode. COPY TT: SAMPLE.S is a host computer command:

```
:TR SAMPLE.S ; S :COPY TT: SAMPLE.S (RET)
:
```

2. To transfer a file to the host computer (in local mode):

```
                                  ← Before entering the TRANSFER command, set the
                                    host computer to be ready to receive data from the
                                    E7000.
:TR FILE.TXT ; S (RET)
:
```

3. To receive a file from the host computer in transparent mode. TYPE FILE.S is a host computer command:

```
:TR FILE.TXT ; R :TYPE FILE.S (RET)
:
```

4. To receive a file from the host computer (in local mode):

```
:TR FILE.TXT ; R (RET)
                                        ← After entering the TRANSFER command, set the host
                                          computer to output data to the E7000.
:
```

9.4.6	VERIFY V	Verifies memory contents against host computer file — Transparent mode and local mode

Command Format

- Verification :VERIFY [Δ<offset>][;[<load module type>][ΔWA]]
  [:<command transferred to host computer>] (RET)

<offset>: Value to be added to the address (can only be specified for a S-type or HEX-type load module)

<load module type>: Load module type

R:	SYSROF-type load module
S:	S-type load module
H:	HEX-type load module
Default:	SYSROF-type load module

WA: Waits for the LF code after a command is transferred to the host computer. Refer to Description below.

<command transferred to host computer>: Specifies a command to be transferred to the host computer (only in transparent mode).

Description

- Verification

— Verifies data transferred from the host computer against data in memory. Verification in different host computer interface modes is described below.

Transparent Mode: After the host computer command below is transferred, the user program from the host computer is verified against the memory contents.

```
:VERIFY;<load module type>:<command transferred to host computer> (RET)
```

<command transferred to host computer>: This command is transferred to the host computer. The characters following the colon (:) are sent directly to the host computer. The command to output the file contents to the terminal must be specified.

The E7000 transfers <command transferred to host computer> and (RET) (CR code: H'0D) to the host computer. At the same time, when echo back display is specified, it displays echo back from the host computer 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 computer. As soon as the E7000 receives the LF code, the E7000 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 computer to no echo and transfer the load module without option WA.

Local Mode: The E7000 does not issue data output requests to the host computer. Therefore, after the VERIFY command is entered, set the host computer to output data.

Remote Mode: Use the INTFC_VERIFY command.

— 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 computer>] (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 internal I/O 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 computer command:

```
:V :COPYLINE F1.ABS TT: (RET)
 VERIFYING ADDRESS 00000000
 <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
                                                computer to output data to the E7000.
 VERIFYING ADDRESS 00000000
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

9.4.7	**INTFC_LOAD** **IL**	**Loads program from host computer** **— Remote mode**

Command Format

- Load :INTFC_LOAD[Δ<offset>][;[<load module type>][ΔN]]:<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)

<load module type>: Load module type

R:	SYSROF-type load module
S:	S-type load module
H:	HEX-type load module
Default:	SYSROF-type load module

N: Specifies not to load <line number symbol>. If omitted, <line number symbol> is loaded.

<file name>: Specifies a file name in the host computer

Description

- Load

— Loads a user program into user memory from the host computer connected in remote mode. Use the H series interface software for the host computer to open the specified file and transfer 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. Enter the unit names of symbols to be defined.

(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.

INTFC_LOAD

Notes

1. The load module cannot be loaded to the internal I/O area.

2. Verification is not performed during load. The program must be verified with the INTFC_VERIFY command if necessary.

3. The LOAD command reloads existing symbols without checking for double definitions to enhance throughput. When reloading the same load module, specify the DEL option or delete existing symbols before executing the INTFC_LOAD command.

Examples

1. To load 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 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 computer — 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 code (H'0A) to the end of each record.
<file name>:	File name in the host computer

Description

- Save

— Saves the specified memory contents in the specified load module type file in the host computer connected in remote mode. Use the H series interface software for the host computer. An S-type or HEX-type load module can be saved. A SYSROF-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.

INTFC_SAVE

Notes

1. Data in the internal I/O area cannot be saved.

2. Verification is not performed. The program must be verified with the INTFC_VERIFY command if necessary. For details, refer to section 9.4.10, INTFC_VERIFY.

Example

To save memory contents in the address range from H'7000 to H'7FFF in host computer 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 computer — Remote mode

Command Format

- Transfer :INTFC_TRANSFER <file name>[;(S/R)][ΔC]:<host computer file name> (RET)

<file name>: Name of file on floppy disk in the E7000
S: Transfer from the E7000 to the host computer
R: Reception from the host computer (default)
C: Inserts CR codes before LF codes in files transferred from the host computer and removes CR codes from files transferred to the host computer
<host computer file name>: File name in the host computer

Description

- Transmission

— Transfers only text files from the E7000 to the host computer connected in remote mode. Use the H series interface software for the host computer.

```
:INTFC_TRANSFER <file name>;S :<host computer file name> (RET)
```

- Reception

— Transfers a file from the host computer connected in remote mode to the E7000. Only text files can be transferred.

```
:INTFC_TRANSFER <file name>;R :<host computer 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 other types of data, an error is generated and the command is aborted.

This command is terminated with EOF (H'1A).

— With a UNIX-based host computer, 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 host computer file SAM.S:

```
:IT SAMPLE.S ;S :SAM.S (RET)
:
```

2. To transfer host computer file FILE.TXT to E7000 file F.T:

```
:IT F.T ;R : FILE.TXT (RET)
:
```

9.4.10	**INTFC_VERIFY** **IV**	**Verifies memory contents against host computer file** **— Remote mode**

Command Format

- Verification :INTFC_VERIFY [Δ<offset>][;<load module type>]:<file name> (RET)

<offset>: Value to be added to the address (can only be specified for an S-type or HEX-type load module)

<load module type>: Load module type

R:	SYSROF-type load module
S:	S-type load module
H:	HEX-type load module
Default:	SYSROF-type load module

<file name>: File name in the host computer

Description

- Verification

— Verifies data transferred from the host computer connected in remote mode against data in memory. Use the H series interface software for the host computer.

```
: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 internal I/O area cannot be verified.

Example

To verify 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 Computer 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 computer connected through the LAN interface.

- Loads a load module file in the host computer to user system memory
- Saves data in the user system memory as a load module file in the host computer
- Transfers files between the E7000 and host computer

The E7000 supports the LAN commands listed in table 10-1 to transfer data between the E7000 and the host computer. 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 computer from the FTP interface (Re-connects the host computer to the FTP interface with the OPEN command)	Usable
FTP	Connects the host computer and 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 computer to be connected by the FTP command	Unusable
LAN_LOAD	Loads a load module file from the host computer to memory via the FTP interface	Unusable
LAN_SAVE	Saves the specified memory contents in the LAN host computer connected via the FTP interface	Unusable
LAN_TRANSFER	Transfers a file between the host computer and E7000	Unusable
LAN_VERIFY	Verifies memory contents against the host computer file	Unusable
LS	Displays the host computer directory connected via the FTP interface	Usable
OPEN	Connects the host computer to the FTP interface	Usable
PWD	Displays the current directory name of the host computer 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 computer 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. 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 computer 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 computer environment, including the host computer name and IP address, to the network database of the host computer. If the operating system of the host computer is UNIX, the host computer environment is specified in the /etc/hosts file. For details, refer to the appropriate host computer'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 computer IP address (host computer connected via FTP interface)
      Specify the name and IP address of the host computer 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 computer name and IP address is written to the system disk. Therefore, the E7000 is automatically connected to the host computer simply by initiating the system disk. The host computer name and IP address can be modified as required.

10.2.2 Data Transfer

Data is transferred by connecting the E7000 to the host computer via the FTP interface after the environmental settings have been completed. In the FTP interface, the optional LAN board supports only the client function. Therefore, the FTP command must be entered to the E7000 and not the host computer to establish FTP interface. Transfer data using the following procedure.

Procedure:

1. Initiate the E7000 system program by using the system disk, on which the host computer name and IP address have been defined by the LAN_HOST command.

2. Connect the E7000 to the designated host computer with the FTP command using the format shown below. Enter the host computer name defined by the LAN_HOST command. In addition, enter the user name and password.

```
:FTP <host computer name> (RET)
 Username: <user name> (RET)
 Password:  <password> (RET)
 login command success
FTP>
```

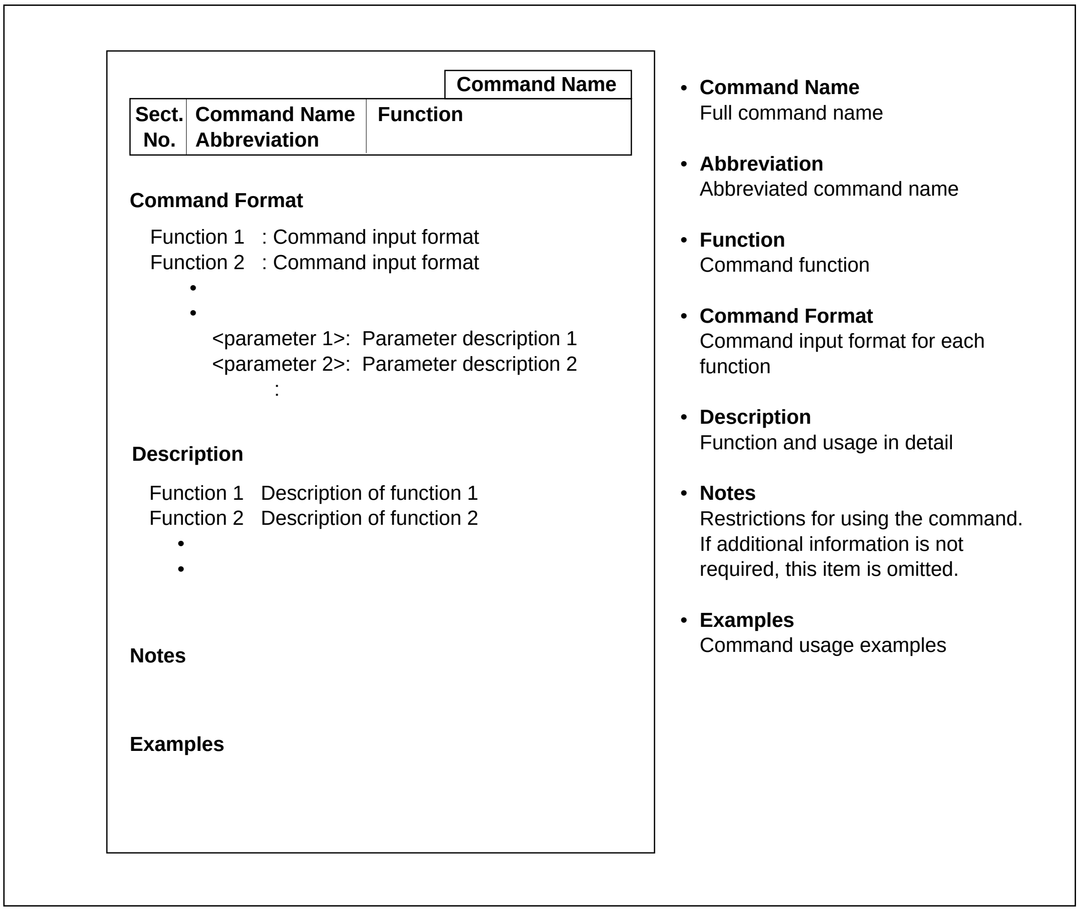
3. Transfer data using the LAN_LOAD, LAN_SAVE, LAN_TRANSFER, or LAN_VERIFY command after the FTP interface is established. For details, refer to the corresponding command descriptions.

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 computer 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:



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**
  Restrictions 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.

ASC		
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 computer, specify the ASCII type with this command. To load a SYSROF-type load module file, binary must be specified with the BIN command. For details, refer to section 10.3.2, BIN.

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_SAVE, LAN_TRANSFER, or LAN_VERIFY 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>
```

BYE		
10.3.3	**BYE** **BYE**	**Terminates the FTP interface**

Command Format

- FTP interface termination : BYE (RET)

Description

- FTP interface termination

  Terminates the FTP interface and changes the prompt to a colon (:). To re-establish the FTP interface, re-enter the FTP command. For details, refer to section 10.3.6, FTP.

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

- Directory modification : CD Δ<directory name> (RET)

    <directory name>: Name of directory to be modified

Description

- Directory modification

    Changes the current directory of the FTP server to the specified directory. The modified directory must be formatted depending on which host computer is connected via the FTP interface.

Example

To change the current directory of the FTP server to subdir:

```
FTP> CD subdir (RET)
 cd command success
FTP>
```

CLOSE

10.3.5	CLOSE CLOSE	Disconnects the host computer from the FTP interface

Command Format

- FTP interface disconnection : CLOSE (RET)

Description

- FTP interface disconnection

  Disconnects the FTP interface from the host computer to which it is currently connected. Before changing host computers, disconnect the FTP interface with this command and re-connect with the OPEN command. For details, refer to section 10.3.14, OPEN.

Example

To disconnect the FTP interface and change the host computer to be connected:

```
FTP> CLOSE (RET)
 bye command success
FTP> OPEN HOST1 (RET)
 username: ABC (RET)
 password: ****** (RET)
 login command success
FTP>
```

10.3.6	**FTP** **FTP**	**Connects host computer and E7000 via the FTP interface**

Command Format

- FTP interface connection : FTP <host computer name> (RET)

  <host computer name>: Name of the LAN host computer to be connected with the FTP server

Description

- FTP interface connection

  — Connects the host computer and E7000 via the FTP interface to enable data transfer with the LAN_LOAD, LAN_SAVE, LAN_TRANSFER, or LAN_VERIFY command. The host computer name specified in this command must be defined with the LAN_HOST command.

  — If <host computer name> matches the host computer 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 computer name>   (RET)
 Username:   (a)    (RET)
 Password:   (b)    (RET)
 login command success
FTP>              (c)
```

(a) Enter user name
(b) Enter password
(c) An FTP> prompt is displayed after FTP connection

Note

A password must be specified before a host computer can be connected via the FTP interface. For a host computer that can login by using only user name, use a login format that requires a password.

FTP

Example

To connect the E7000 to host computer 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
:
```

LAN_HOST		
10.3.8	**LAN_HOST** **LH**	**Specifies, modifies, and displays the name and IP address of the host computer to be connected by the FTP command**

Command Format

- Specification and modification : LAN_HOST;S (RET)

- Display : LAN_HOST (RET)

Description

- Specification and modification

  — Specifies the name and internet (IP) address of the host computer to be opened with the FTP command. A maximum of nine names and internet addresses can be specified.

  — The specified host computer name and IP address can be modified in interactive mode as shown below. After displaying the specified host computer names and IP addresses, the E7000 waits for the selection number input. After the selection number is entered, the E7000 waits for the host computer name and IP address inputs. 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>   (a)
 01      xxxxxx           xx.xx.xx.xx    02      xxxxxx           xx.xx.xx.xx
 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 computer name and IP address currently defined. If nothing is specified, displays a space. NO indicates selection number.

(b) Enter the selection number (1–9) to be set or modified.

(c) Displays the host computer name for the specified selection number. To specify a new host computer name, enter the host computer name using 15 or less characters. To delete the old host computer 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 computer 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 computer name and IP address settings are displayed.

— Specified host computer 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 computer.

— Before executing the FTP command, use the LAN_HOST command to specify the name and internet address of the host computer that is 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 computer is defined, the name and IP address of the host computer need not be specified.

LAN_HOST

- Display

  Displays the LAN host computer names and internet addresses specified in the LANCNF.SYS file of the E7000 system disk.

  :LAN_HOST (RET)

Examples

1. To add a host computer to be connected via the FTP 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 computer 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 computer 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
:
```

LAN_LOAD		
10.3.9	**LAN_LOAD** **LL**	**Loads a load module file from the host computer to memory via the FTP interface**

Command Format

- Load : LAN_LOAD [Δ<offset>][;[<load module type>][ΔN]:<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)

<load module type>: Load module type

R:	SYSROF-type load module
S:	S-type load module
H:	HEX-type load module
Default:	SYSROF-type load module

N: Specifies that <line number symbol> is not to be loaded. If omitted, <line number symbol> information is loaded.

<file name>: A LAN host computer file name

Description

- Load

— Loads a load module file from the host computer to memory via the FTP interface. Before executing this command, the E7000 must be connected to the host computer with the FTP command.

— The current load address is displayed in the format below.

```
LOADING ADDRESS   xxxxxxxx
```

xxxxxxxx: The 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.

```
: LAN_LOAD  <offset>; S:<file name>    (RET)
```

If an offset is specified, the load address is calculated as follows:

Load address = <load module address> + <offset>

— 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 symbols are to be selected.
Y: Loads all symbols.
N: Enables the selection of symbols by unit names.
If Y is entered, the confirmation request messages (b) are not displayed. If N is entered, the confirmation request messages are displayed. Enter unit names of symbols to be loaded.

(b) Specifies a unit name. Loading starts when the period (.) is entered as a response to a confirmation request message. Up to ten unit names can be specified.

— If option N is specified, <line number symbol> information among the symbol information for the SYSROF-type load module is not loaded.

LAN_LOAD

Notes

1. A load module file cannot be loaded to the internal I/O area.

2. Verification is not performed during load. The program must be verified with the LAN_VERIFY command, if necessary. For details, refer to section 10.3.12, LAN_VERIFY.

3. To enhance throughput, the LAN_LOAD command reloads existing symbols 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. For details, refer to section 10.3.2, BIN.

Example

To load a SYSROF-type load module, enter the following command line. F11.ABS indicates the host computer file name. Before entering the LL command, connect the E7000 to the host computer with the FTP command:

```
:FTP HOST1 (RET)
 Username: USER1 (RET)
 Password: ******** (RET)
 login command success
FTP> LL :F11.ABS (RET)
 ALL SYMBOL LOAD (Y/N) ? N (RET)
 LOAD UNIT NAME (name/.) ? un001 (RET)
 LOAD UNIT NAME (name/.) ? . (RET)
 LOADING ADDRESS   00007000
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
FTP>
```

10.3.10	LAN_SAVE LSV	**Saves the specified memory contents in the LAN host computer 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
	Default: S-type load module
LF:	LF (H'0A) is added to the end of each record.
<file name>:	A LAN host computer file name

Description

- Save
  - — Saves the specified memory contents in the host computer 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 computer with the FTP command.
  - — The current save address is displayed as follows:

```
SAVING ADDRESS  xxxxxx

      xxxxxx: 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 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 internal I/O area cannot be saved.

2. Verification is not performed. Verify the program with the LAN_VERIFY command, if necessary. For details, refer to section 10.3.12, LAN_VERIFY.

Example

To save the memory contents in the address range from H'7000 to H'7FFF in the host computer as an S-type load module file (file name: F11.S), enter the following command line. Before entering the LSV command, connect the E7000 to the host computer with the FTP command:

```
FTP>LSV 7000 7FFF :F11.S (RET)
 SAVING ADDRESS   00007000
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
FTP>
```

10.3.11	LAN_TRANSFER LTR	Transfers a file between the host computer 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 computer
  R: Reception from the host computer (Default)
  <file name2>: A host computer file name

Description

- Transfer

  Transfers and receives file between the E7000 and host computer via the FTP interface. Before entering this command, the following steps must be completed.

  (1) The E7000 must be connected to the host computer with the FTP command (section 10.3.6).

  (2) The file type must be specified as either binary or ASCII with the BIN (section 10.3.2) or ASC (section 10.3.1) command, respectively.

  — Transmission

  Transfers files from the E7000 to the LAN host computer via the FTP interface.

  : LAN_TRANSFER <file name1> ; S :<file name2> (RET)

  — Reception

  Transfers files from the host computer 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.

Note

Before executing this command to transfer the command chain file created by the host computer, specify the file type as ASCII with the ASC command.

Examples

1. To transfer file SAMPLE.S from the E7000 to file TEST.S on the host computer, enter the following command lines. Before executing the LTR command, the E7000 must be connected to the host computer 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 computer to file E7.CC on the E7000, enter the following command lines. 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 computer file**

Command Format

- Verification : LAN_VERIFY [Δ<offset>][;<load module type>]:<file name> (RET)

<offset>: (1) Value to be added to the address (can be specified only for an S-type or HEX-type load module)
(2) Stard address (for M type)
Default: 0
<load module type>: Load module type
R: SYSROF-type load module
S: S-type load module
H: HEX-type load module
Default: SYSROF-type load module
<file name>: A LAN host computer file name

Description

- Verification

— Verifies file transferred from the host computer connected via the FTP interface against data in memory in the following format. Before executing this command, connect the E7000 to the host computer with the FTP command.

FTP > LAN_VERIFY <load module type>:<file name> (RET)

— If a verification error occurs, the address and its contents are displayed as follows:

```
<ADDR>      <FILE>      <MEM>
xxxxxxxx     yy 'y'       zz 'z'
```

xxxxxxxx: Verification error address
yy 'y': Load module data (in hexadecimal and ASCII character)
zz 'z': Memory data (in hexadecimal and ASCII character)

LAN_VERIFY

— 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.

FTP > LAN_VERIFY <offset> ; S [:<file name>] (RET)

If an offset is specified, the load address is calculated as follows:

Load address = <load module address> + <offset>

Notes

1. Symbolic information cannot be verified.

2. Data in the internal I/O 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. For details, refer to section 10.3.2, BIN.

Example

To verify SYSROF-type load module file F11.ABS against the memory contents:

```
:FTP HOST1 (RET)
 Username: USER1 (RET)
 Password: ******** (RET)
 login command success
FTP>LV :F11.ABS (RET)
 VERIFYING ADDRESS   00000000
 TOP ADDRESS = 00000000
 END ADDRESS = 000000FF
FTP>
```

10.3.13	**LS** **LS**	**Displays the host computer directory connected via the FTP interface**

Command Format

- Display : LS [Δ <directory name>] (RET)

  <directory name>: Name of host computer directory
  (Default: Current directory of the host computer)

Description

- Display

  Displays the specified directory in the host computer connected via the FTP interface. If <directory name> is omitted, the current directory contents are displayed. Note that the directory name must be specified according to the connected host computer format.

Example

To display the contents of current directory of the host computer:

```
FTP>LS (RET)
abc.s
xyz
FTP>
```

OPEN		
10.3.14	**OPEN** **OPEN**	**Connects the host computer to the FTP interface**

Command Format

- FTP interface connection : OPEN <host computer name> (RET)

<host computer name>: Name of host computer to be connected via the FTP interface
(The host computer name must have been defined with the LAN_HOST command)

Description

- FTP interface connection

Connects the E7000 to the specified host computer via the FTP interface. This command can also be used to change the host computer to be connected to the E7000. To change the host computer correctly, first disconnect the current host computer by using the CLOSE command and then connect the new host computer by using this command.

```
FTP>OPEN <host computer name> (RET)
 Username: (a) (RET)
 Password:  (b) (RET)
 login command success
FTP>
```

(a): Enter user name
(b): Enter password

Note

A password must be specified before a host computer is connected via the FTP interface. For a host computer that can login by using only user name, use a login format that requires a password.

Example

To disconnect the E7000 from the current host computer and connect it to the new host computer HOST1:

```
FTP>CLOSE (RET)
 bye command success
FTP>OPEN HOST1 (RET)
 Username: USER1 (RET)
 Password: ******** (RET)
 login command success
FTP>
```

PWD		
10.3.15	**PWD** **PWD**	**Displays the current directory name of the host computer connected via the FTP interface**

Command Format

- Display : PWD (RET)

Description

- Display

  Displays the current directory name of the host computer connected via the FTP interface.

Example

To display the current directory name of the host computer connected via the FTP interface:

```
FTP>PWD (RET)
 /usr/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_SAVE, LAN_TRANSFER, or LAN_VERIFY 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>
```

LOGOUT		
10.3.17	**LOGOUT** **LO**	**Disconnects from the TELNET**

Command Format

- TELNET disconnection : LOGOUT (RET)

Description

- TELNET disconnection

  Disconnects the E7000 from the TELNET. This command is valid only when the E7000 is connected to the host computer 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 computer (IBM PC) can be performed by the commands listed in table 11-1. For details on these commands, refer to section 11.3, E7000PC-Related Data Transfer Commands.

- Loads a load module file in the host computer to user system memory
- Saves data in the user system memory as a load module file in the host computer
- Transfers files between the E7000 and host computer (IBM PC)

Table 11-1 E7000PC-Related Data Transfer Commands

Command Name	E000PC Command Use
LOAD	Loads program from host computer (IBM PC)
SAVE	Saves program in host computer (IBM PC)
VERIFY	Verifies memory contents against host computer file (IBM PC)

11.2 E7000PC and IBM PC Connection

The flow of data transfer between the E7000PC and IBM PC is shown in figure 11-1.

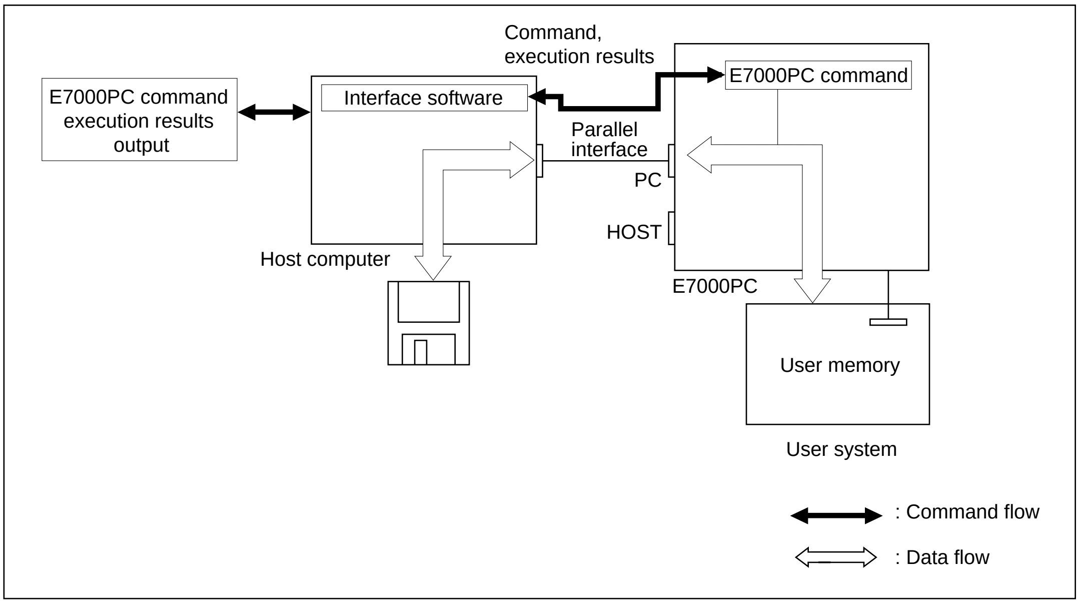


Figure 11-1 Data Transfer Flow

Procedure:

Start up host computer.

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 computer. Emulator commands can now be entered from the host computer.

Data transfer from host computer (IBM PC) to E7000PC:

Execute E7000PC data receive command.

The E7000PC data receive command (LOAD or VERIFY) can transfer data from the host computer to the E7000PC.

Example:

```
LOAD:<host computer file name>
```

Data transfer from E7000PC to host computer (IBM PC):

Execute E7000PC data transmission command.

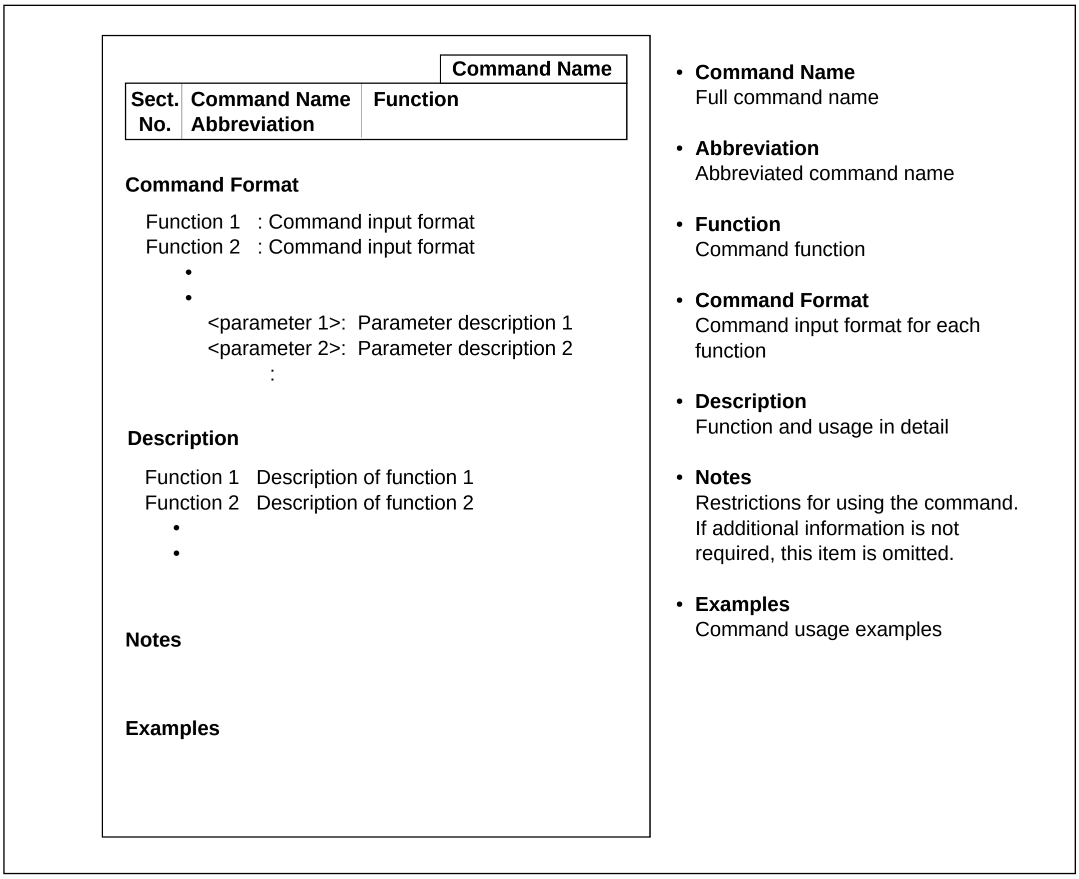
The E7000PC data transmission command (SAVE) can transfer data from the E7000PC to the host computer.

Example:

```
SAVE 0 1FFF:<host computer file name>
```

11.3 E7000PC-Related Data Transfer Commands

This section provides details of host-computer related commands using the format shown in figure 11-2.



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**
  Restrictions 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.

LOAD

11.3.1	**LOAD** **L**	**Loads program from host computer (IBM PC)**

Command Format

- Load :LOAD[Δ<offset>][;[<load module type>][ΔN]]:<file name> (RET)

<offset>: Value to be added to the address (can be specified only for an S-type or HEX-type load module)

<load module type>: Load module type

S:	S-type load module
H:	HEX-type load module
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.

<file name>: Specifies a file name in the host computer (IBM PC).

Description

- Load

— Loads a user program into user memory from the host computer. Use the H-series interface software for the host computer to open the specified file and transfer 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 a 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.

Notes

1. Data cannot be loaded to the internal I/O area.

2. Verification is not performed. The program must be verified with the VERIFY command if necessary.

3. To enhance throughput, the LOAD command reloads existing symbols 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 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)
 LOADING ADDRESS 00007000
 TOP ADDRESS = 00007000
 END ADDRESS = 00007A3F
:
```

2. To load S-type load module ST.MOT:

```
:L ;S :ST.MOT(RET)
 LOADING ADDRESS 00000000
 TOP ADDRESS = 00000000
 END ADDRESS = 00003042
:
```

11.3.2	**SAVE** **SV**	**Saves program in host computer (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 code (H'0A) to the end of each record.

<file name>: File name in the host computer (IBM PC)

Description

- Save

— Saves the specified memory contents in the specified load module type file in the host computer. Use the H series interface software for the host computer. An S-type or HEX-type load module can be saved. A SYSROF-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.

SAVE

Notes

1. Data in the internal I/O area cannot be saved.

2. Verification is not performed. Verify the program with the VERIFY command if necessary. For details, refer to section 11.3.3, VERIFY.

Example

To save memory contents in the address range from H'7000 to H'7FFF in host computer file F11.MOT in S-type load module format:

```
:SV 7000 7FFF :F11.MOT (RET)
 SAVING ADDRESS 00007000
 TOP ADDRESS = 00007000
 END ADDRESS = 00007FFF
:
```

11.3.3	VERIFY V	Verifies memory contents against host computer 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)
<load module type>: Load module type
S: S-type load module
H: HEX-type load module
R: SYSROF-type load module
Default: SYSROF-type load module
<file name>: File name in the host computer

Description

- Verification

— Verifies data transferred from the host computer against data in memory. Use the H series interface software for the host computer.

```
: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'
```

xxxxxxxxx: 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 computer> (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 internal I/O area cannot be verified.

Example

To verify SYSROF-type load module F1.ABS against the memory contents:

```
:V :F1.ABS  (RET)
 VERIFYING ADDRESS 00000000
 <ADDR>     <FILE>    <MEM>
 00001012    31'1'     00'.'
 TOP ADDRESS = 00000000
 END ADDRESS = 00003FFF
:
```

Section 12 Error Messages

12.1 E7000 Error Messages

The E7000 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)	An error occurred in the E7000 system program or station. Error code nn gives specific details. Contact a Hitachi sales agency and inform them of the code and statement.
2	HOST I/O ERROR (nn)	An I/O error occurred between the emulator and host computer. Error code nn gives specific details. Refer to table 12-2.
3	FD I/O ERROR (nn)	An 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 E7000 system program or an error occurred in the connection between the emulator pod and E7000 station. Check the E7000 system program and emulator pod type numbers, and check the connection between the E7000 station and the emulator pod.
6	USER SYSTEM NOT READY	The user clock or crystal oscillator clock was not input and therefore could not be selected. The emulator internal clock was used instead. Check if the clock signal is output correctly.
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	A CRC error occurred during disk read/write. Reformat or use another floppy disk.
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 the specified file name.

Table 12-1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
14	INVALID FILE NAME	The file name has an invalid 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	The 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	The file to be copied is too large. The maximum size allowed is shown as xxxxx. 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	The 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	The item has already been defined. Delete the existing item and redefine it.
29	CC COMMAND IN COMMAND FILE	The command file contains a COMMAND_CHAIN command which cannot be used. Delete the COMMAND_CHAIN command from the file.
30	SYMBOL IN USE	The specified symbol cannot be deleted because it is used in a BREAK, BREAK_SEQUENCE, BREAK_ CONDITION, LEC, TRACE_MEMORY, or TRACE_CONDITION command. Clear the symbol in that command, and delete it again.

Table 12-1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
31	INSUFFICIENT MEMORY	The size of emulation memory to be allocated specified with the MAP command was not available. Emulation memory was allocated within the available memory size.
32	INVALID ASM MNEMONIC	An instruction mnemonic in an assembly sentence is invalid. Correct the instruction mnemonic.
33	INVALID ASM OPERAND	An operand in an assembly statement is invalid. Correct the operand.
34	ALREADY ASSIGNED	The specified printer or file has already been assigned. Cancel the assignment and re-assign with the PRINT command.
35	CAN NOT USE THIS MODE	• GO The 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. When loading the same program, this error message is displayed because the emulator does not check double definition of the symbol. Delete and reregister a symbol.
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 The trace buffer is empty. Check trace conditions and execution state, and re-execute. Then display trace information.
41	NO OPTION BOARD OR LED_BOX DISCONNECT	There is no optional memory board or LED_BOX connected. Connect them and execute the command again.
42	GUARDED I/O WRITE	Writing to a special internal I/O register that can only be accessed by the emulator was attempted with the MEMORY command. Check the address. Writing to the register does not affect user program operation.
44	VERIFY ERROR	Writing to ROM was attempted or there was a memory error during verification. Check memory.

Table 12-1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
45	NOT FOUND	The specified data or information was not found. Correctly specify data.
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.
49	ILLEGAL INSTRUCTION ADDRESS	The memory contents of the specified address of BREAK and BREAK_SEQUENCE commands is break instruction (H'5770). No data can be set to this address.
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 I/O AREA	An attempt was made to access the internal I/O area. This area cannot be accessed with this command. Check the specified address.
53	LABEL TABLE OVERFLOW	The assemble label table overflowed. Reduce the declaration and reference points.
54	INVALID CONFIGURATION FILE	The configuration file on the E7000 system disk contains invalid data. Initiate the emulator with a correct configuration file.
55	CONFIGURATION FILE NOT FOUND	The configuration file was not found on the E7000 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 which has been loaded to the emulator. The warm start must be performed for a file which has been loaded to the emulator. Check the configuration file.
59	TOO MANY CHARACTERS	Too many characters were specified. Check the number of characters.
61	CAN NOT GET INTO PARALLEL MODE	The execution mode specified with the GO command prevents the emulator from entering parallel mode. Change the execution mode.
62	LAN BOARD DISCONNECT	This command cannot be executed because the LAN board is not installed. Install the optional LAN board and re-enter the command.

Table 12-1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
66	OUT OF COVERAGE PAGE	An attempt was made to display coverage information on an address out of the area specified by the BS option of the EXECUTION_MODE command. Check the address.
67	LAN I/O ERROR	An attempt was made to access the 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	STOPPED THE BACKGROUND INTERRUPT	At E7000 initiation: The loop program address specified in the BACKGROUND_INTERRUPT command is not in the RAM area. Correct the loop program address. In user interrupt enabled state while command input is being waited: A break has occurred during user interrupt processing, and the loop program for accepting user interrupts has stopped. Modify and reload the interrupt processing routine.
70	MAPPING BOUND MUST 128KB	Memory was allocated in 128-kbyte units with the MAP or MOVE_TO_RAM command. For details, refer to the MAP command.
71	MAPPING BOUND MUST 64KB	Memory was allocated in 64-kbyte units with the MAP or MOVE_TO_RAM command. For details, refer to the MAP command.
72	MAPPING BOUND MUST 1MB	The memory was allocated in 1-Mbyte 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 were 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 before GO command execution. A hardware error might have occurred or the contents of the memory address may be a break instruction (H'5770). Correct the error, and reload and re-execute the program. (Warning message)

Table 12-1 Emulator Error Messages (cont)

Error No.	Error Message	Description and Solution
77	ALL BREAK POINTS DELETED	All software breakpoints specified by the BREAK or BREAK_SEQUENCE command were cancelled. (Warning message)
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.
79	RUN_TIME OVERFLOW	Execution time measured with the PERFORMANCE_ANALYSIS command overflows.
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 by the assembler. Processing was initiated from the odd address.
83	INVALID OPERAND SIZE	An invalid operand size was specified by the assembler. Processing was performed with the correct size.
84	INVALID ABSOLUTE ADDRESS	An invalid address was specified by the assembler. Processing was performed with the maximum address allowed.
85	COVERAGE INITIALIZED	All coverage information was cleared because the BS option specification of the EXECUTION_MODE command was changed.
86	INTERNAL I/O AREA	The internal I/O area is included in the processing range. Commands other than MEMORY cannot be performed on the internal I/O area.
87	PERFORMANCE_ANALYSIS TABLE BUSY	The measurement time minimum unit cannot be changed during execution time measurement.
88	INTERNAL ROM AREA	The contents of memory area other than internal ROM was transferred by the MOVE_TO_RAM command.

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 computer specifications.
D2	Overrun error: The E7000 control method is not recognized by the host computer. Refer to the description of control methods in section 13.4.1, Control Methods.
D3	Framing error: The baud rate and stop bit specified with the HOST command must match the host computer 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 computer. Also check the operation status of the host computer.
B0	System program 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 E7000 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 system program.
02	A non-existent disk drive is specified. Reload and re-initiate the E7000 system program.
03	An invalid sector number was accessed. Reload and re-initiate the E7000 system program.
05	The next command was issued during FDC command execution. Reset the E7000 by switching it off and on.
07	The file attribute is READ ONLY: Data cannot be written to this file. Remove the write protection or change floppy disks.
0A	The floppy disk directory area is full. Use a new floppy disk.
11	No floppy disk is inserted. Insert a floppy disk.
12	The floppy disk is write-protected. Remove the write protection.
21	Data transfer between a floppy disk and memory failed. Re-enter the command.
32	The sector to be accessed was not found. Re-enter the command.
33	Deleted Data Mark was detected. Reformat the floppy disk.
41	An error occurred during head transfer. Re-enter the command.
51	The floppy disk remains in the busy state. Reset the E7000 by switching it off and on.
52	A FAULT signal was sent 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	An FDC error occurred. 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	A DMAC error occurred. Reset the E7000 by switching it off and on.
C1	The record is too long to 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 it with a new floppy disk.

Table 12-4   Floppy Disk Error Messages

Message	Description and Solution
*** FD NOT READY	No floppy disk is inserted. 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 floppy disk.
*** FD CRC ERROR	A CRC error occurred during read/write to the floppy disk. Reformat or replace the floppy 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 (refer to 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 computer error messages.

Table 12-5   LAN I/O Error Messages

Error No.	Error Message	Description
01	not listen	The socket cannot be created.
02	Insufficient Buffer	The internal buffer is insufficient.
03	Socket not Support	The requested function is not supported.
04	Socket is Already	The socket has already been connected.
05	time out error	A timeout error has occurred.
06	Ip Address Nothing	The IP address destination is undefined.
07	Not Socket Connection	The socket has not been connected.
08	connection 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 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 Messages

Error No.	Process
01	Initialization
02	TELNET data transfer
03	TELNET close
04	TELNET open
10	FTP connection
20	File transmission
30	File reception
40	FTP disconnection
50	Directory modification
60	Directory display
70	Current directory display
80	File transfer binary specification
90	File transfer ASCII specification
A0	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 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 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 retransfer.
INTFC ERROR - I/O ERROR	An I/O error has occurred during file transfer. Check the cable connection and the operating environment, and retransfer.
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 PC 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 IBM PC expansion slot. Check the DIP switch setting on the interface board and that the interface board is inserted in the expansion slot correctly, and 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.

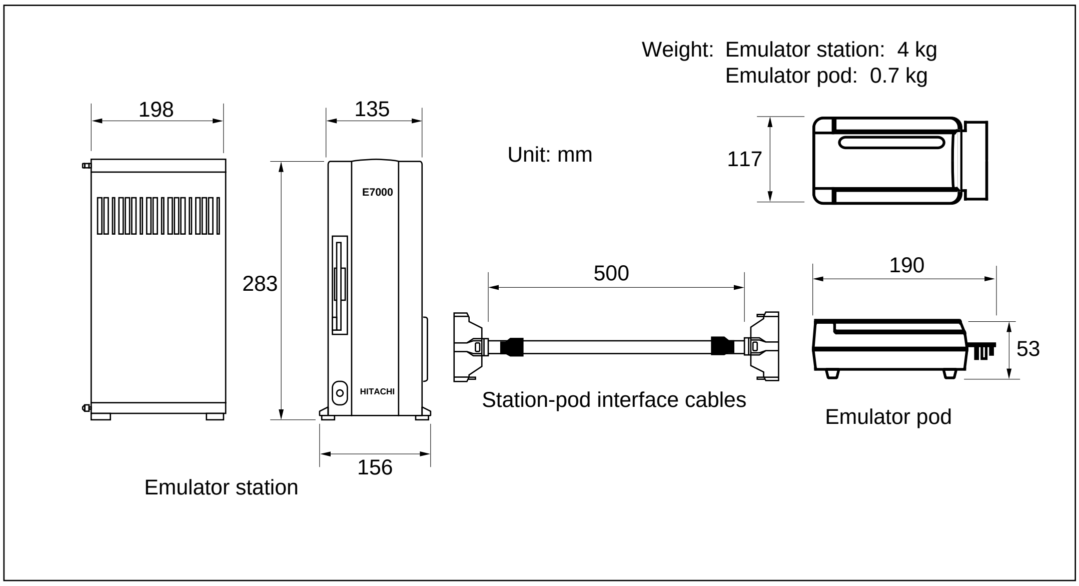


Figure A-1   E7000 External Dimensions and Weight

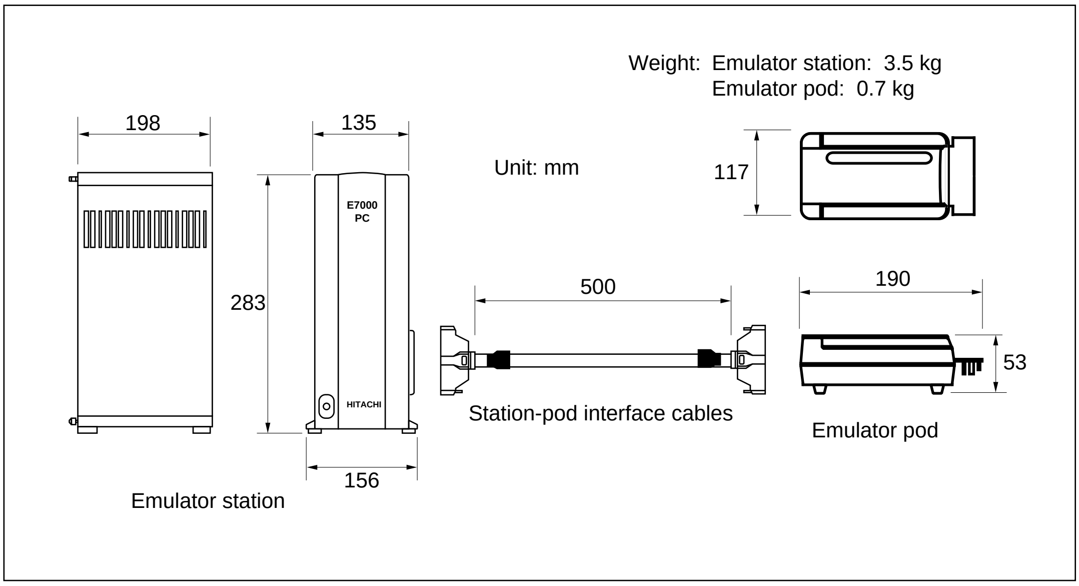
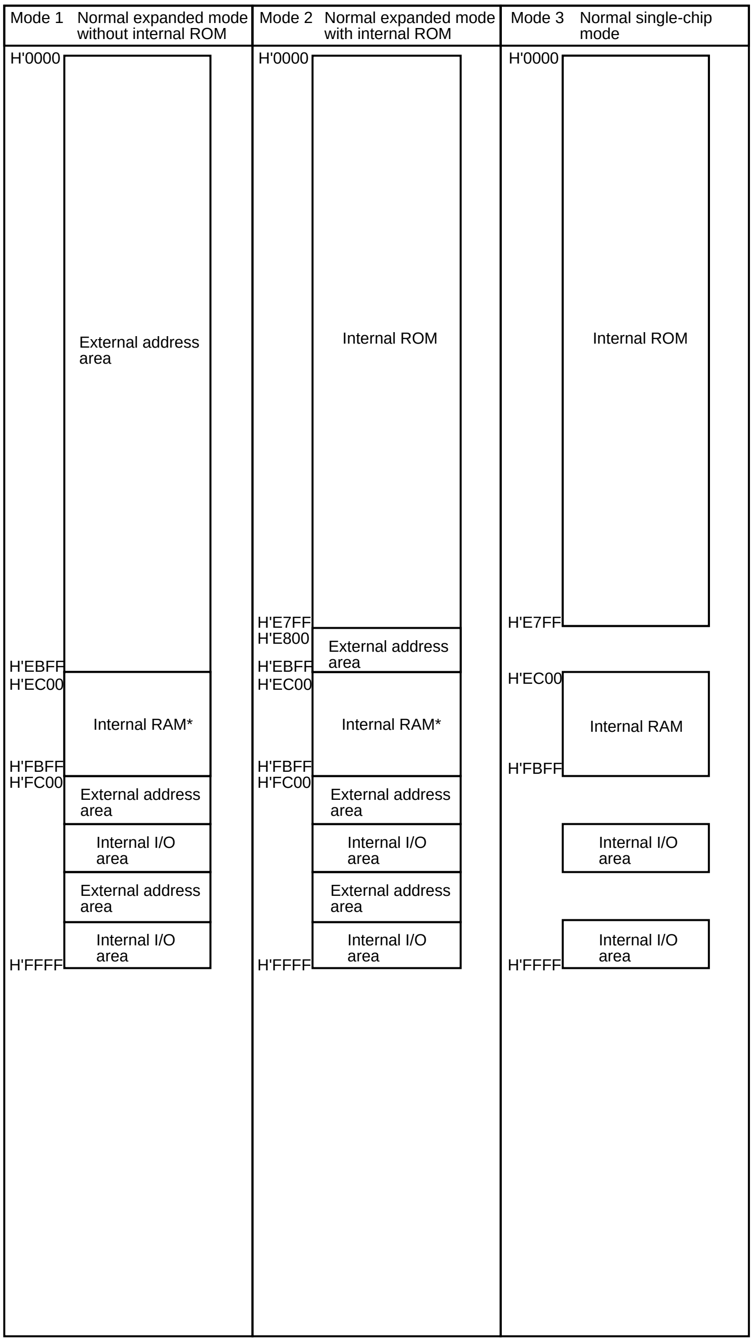


Figure A-2   E7000PC External Dimensions and Weight

Appendix B Memory Map

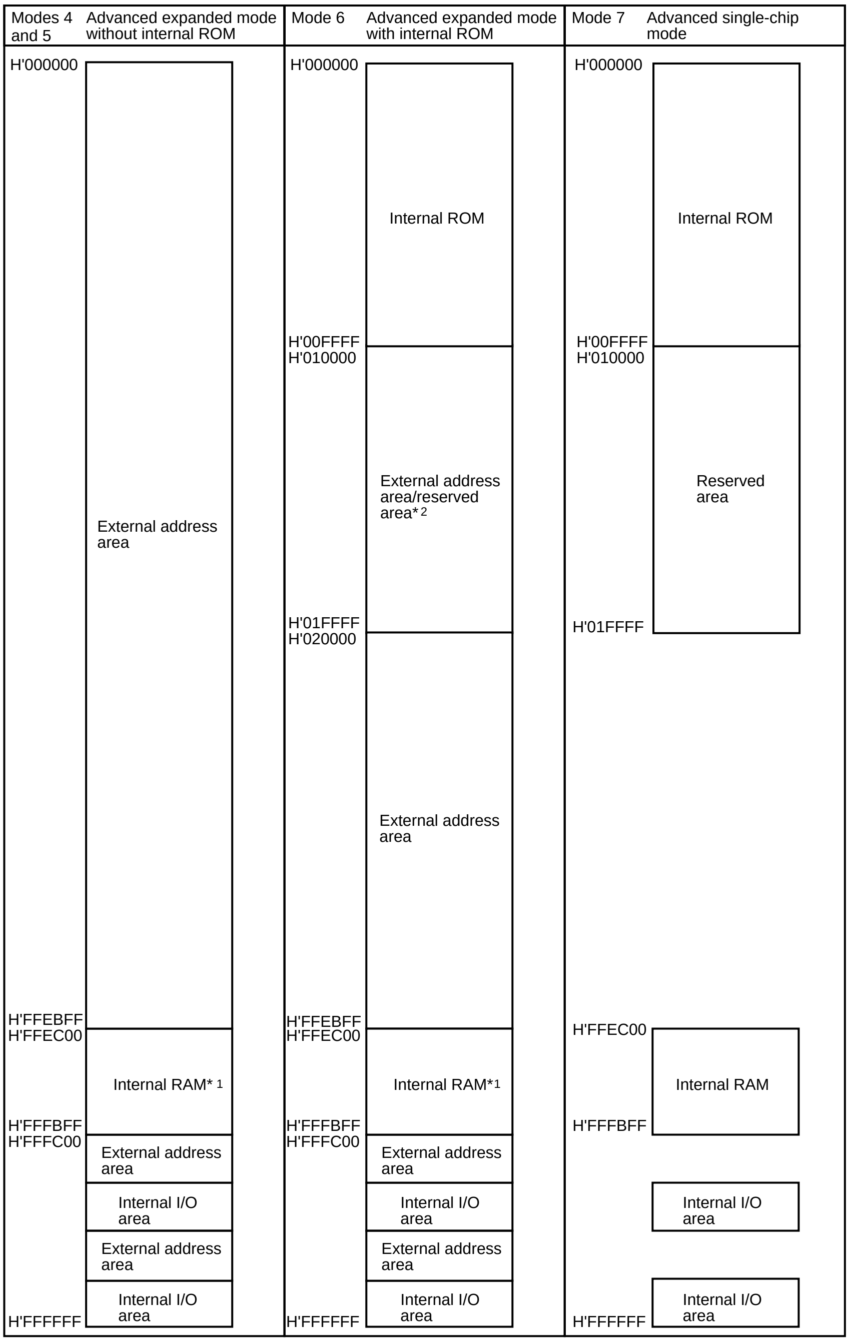
The MCU operating mode can be selected from seven modes (modes 1 to 7) by combining single-chip/expanded, internal ROM enabled/disabled, and expanded data bus width.

After reset, the settings of the mode pins (MD2 to MD0) or the MODE command settings determine the operating mode. The expanded data bus width is determined depending on the attribute of the area in the address space specified by the bus controller. In each operating mode, the address space and pin functions change.



Note: This area can be changed to an external address area by clearing the RAME bit of the SYSCR in the MCU to 0.

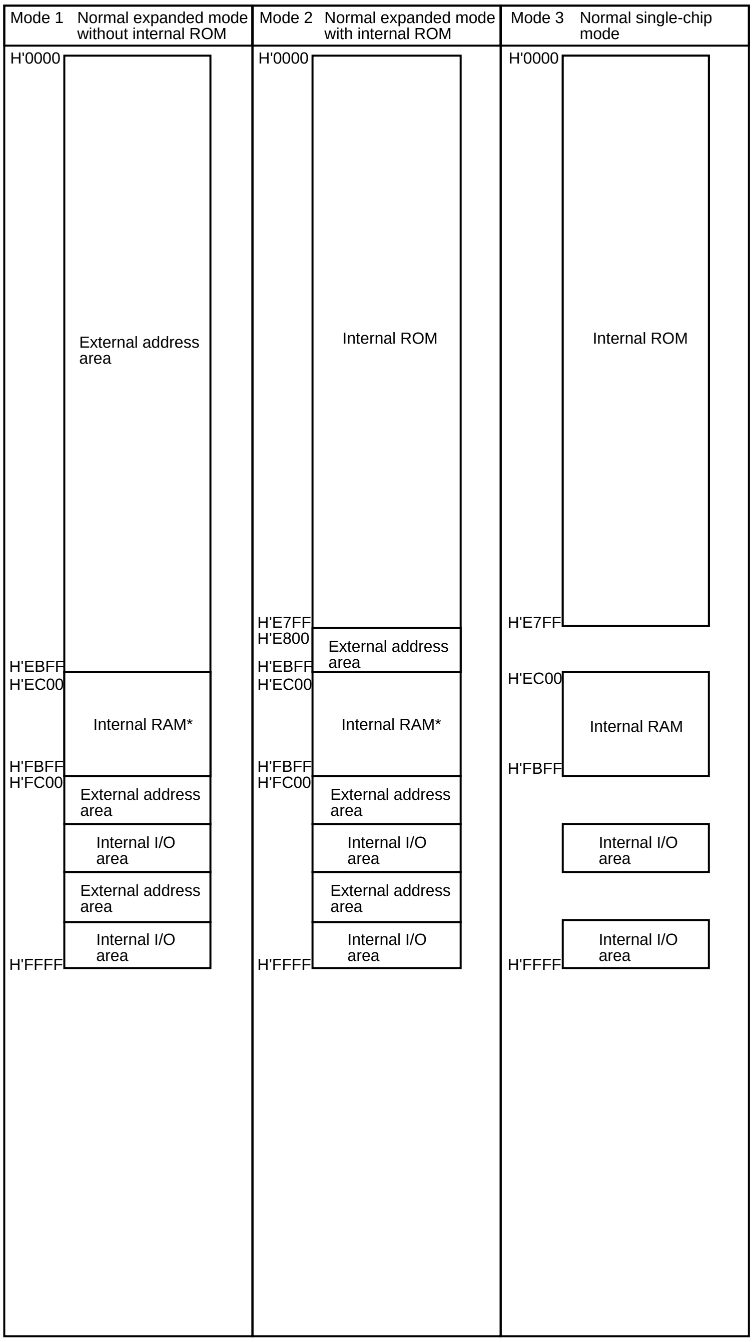
Figure B-1 H8S/2653 Memory Map



Notes: 1. This area can be changed to an external address area by clearing the RAME bit of the SYSCR in the MCU to 0.

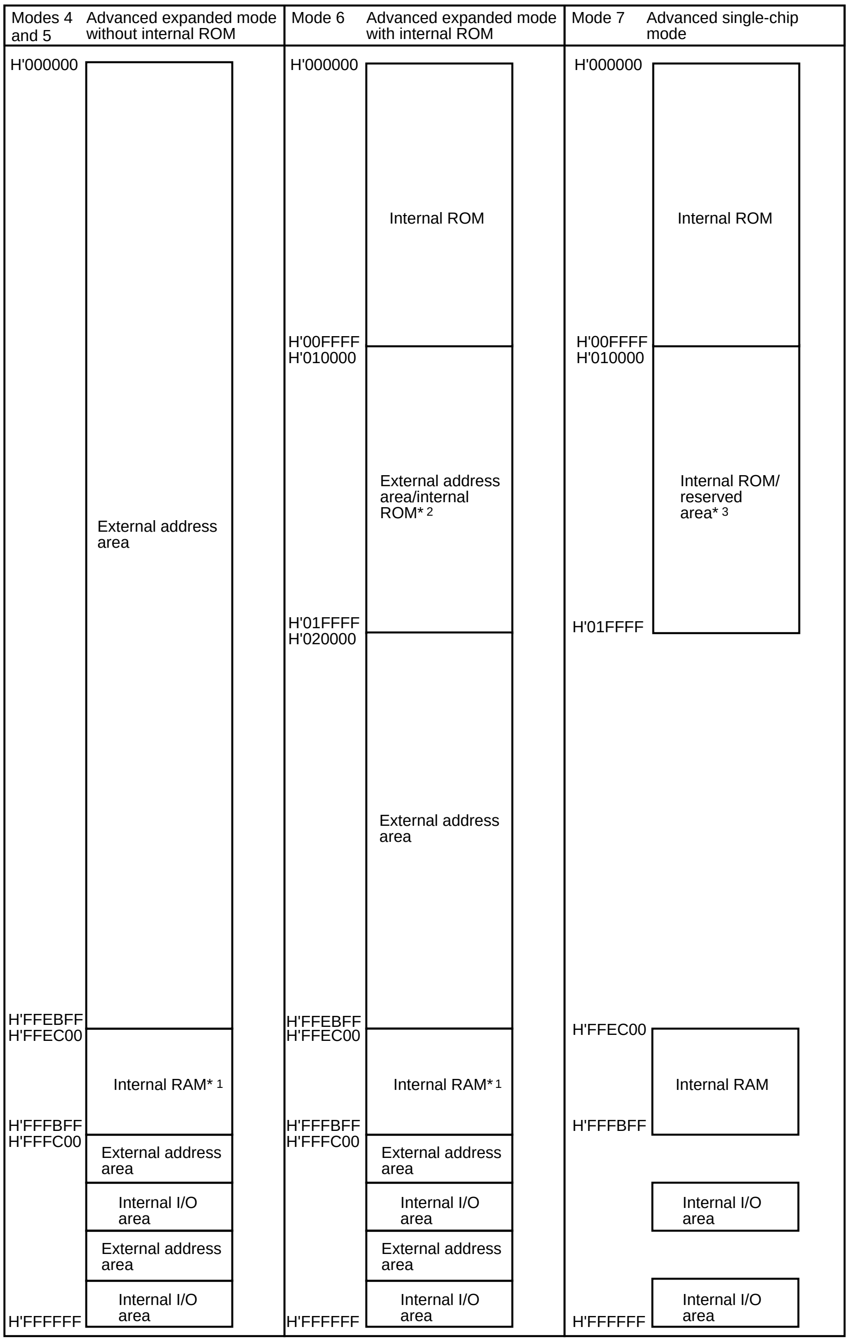
2. This area is an external address area when the EAE bit of the BCRL in the MCU is 1 and a reserved area when the EAE bit is 0.

Figure B-1 H8S/2653 Memory Map (cont)



Note: This area can be changed to an external address area by clearing the RAME bit of the SYSCR in the MCU to 0.

Figure B-2 H8S/2655 Memory Map



Notes: 1. This area can be changed to an external address area by clearing the RAME bit of the SYSCR in the MCU to 0.
2. This area is an external address area when the EAE bit of the BCRL in the MCU is 1 and an internal ROM when the EAE bit is 0.
3. This area is a reserved area when the EAE bit of the BCRL in the MCU is 1 and an internal ROM when the EAE bit is 0.

Figure B-2 H8S/2655 Memory Map (cont)

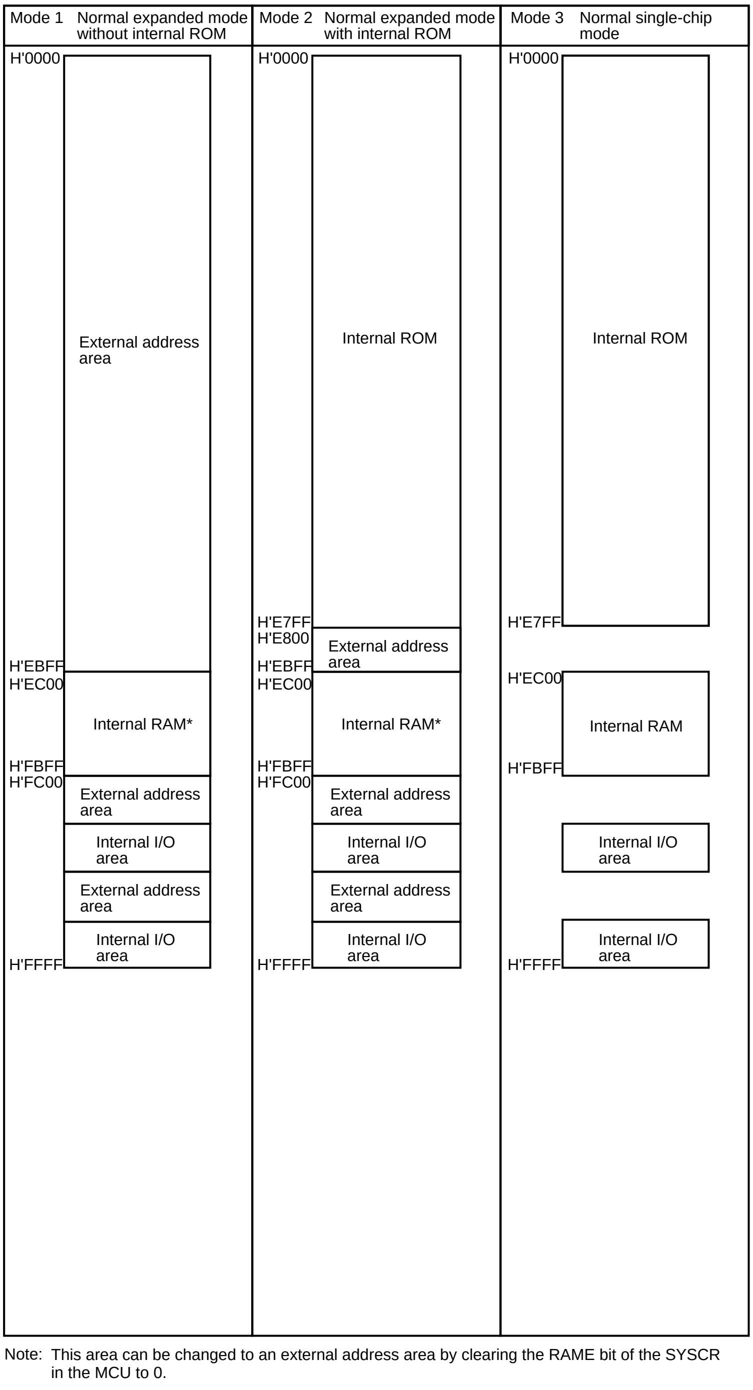
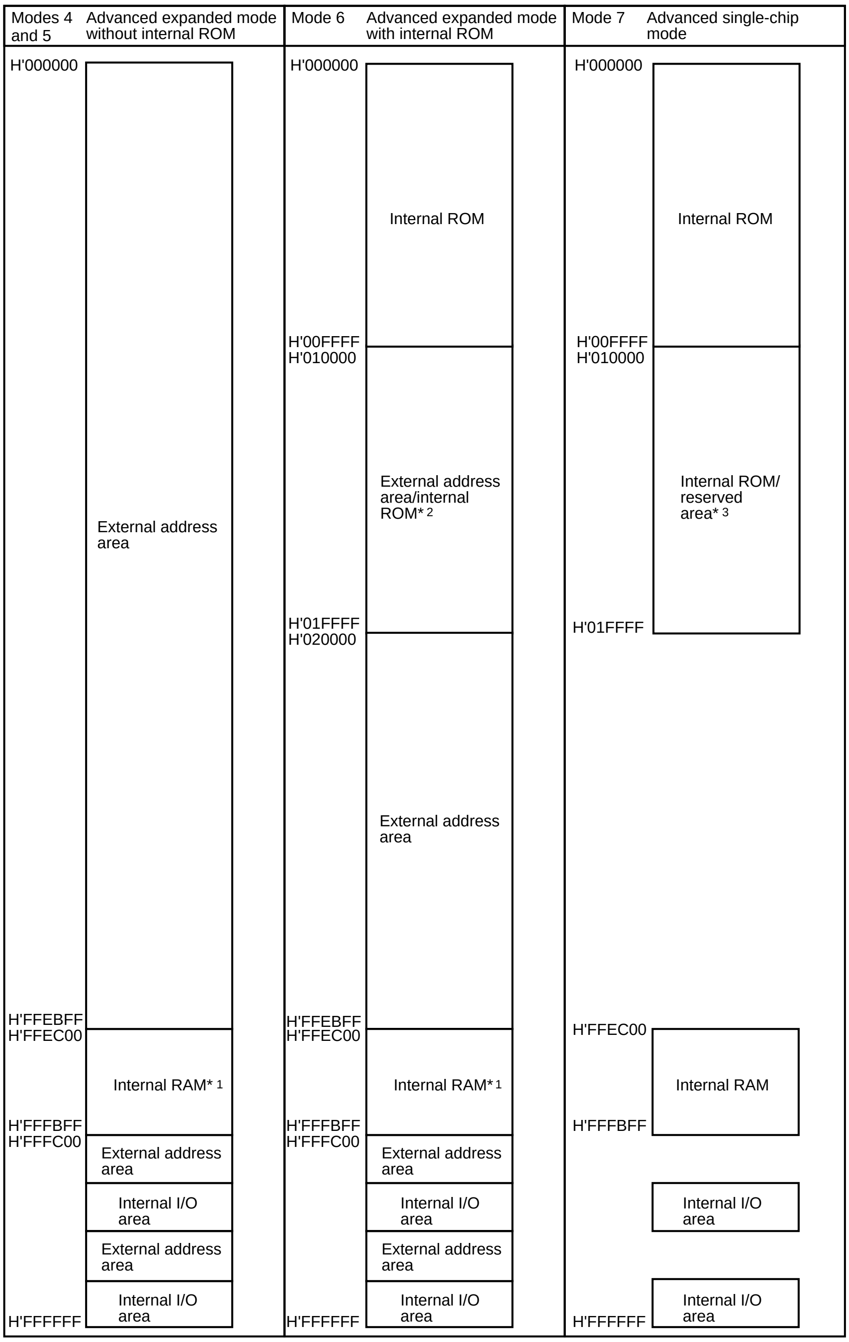


Figure B-3 H8S/2245 Memory Map



Notes: 1. This area can be changed to an external address area by clearing the RAME bit of the SYSCR in the MCU to 0.
2. This area is an external address area when the EAE bit of the BCRL in the MCU is 1 and an internal ROM when the EAE bit is 0.
3. This area is a reserved area when the EAE bit of the BCRL in the MCU is 1 and an internal ROM when the EAE bit is 0.

Figure B-3 H8S/2245 Memory Map (cont)

Appendix C Precautions on User System Connection

When connecting the emulator pod to the user system, note the following:

1. Secure the emulator station location.
   Place the emulator station and emulator pod so that the trace cable is not bent or twisted, as shown in figure C-1. A bent or twisted cable will impose stress on the user interface leading to connection or contact failure. Make sure that the emulator station is placed in a secure position so that it does not move during use and impose stress on the user interface.

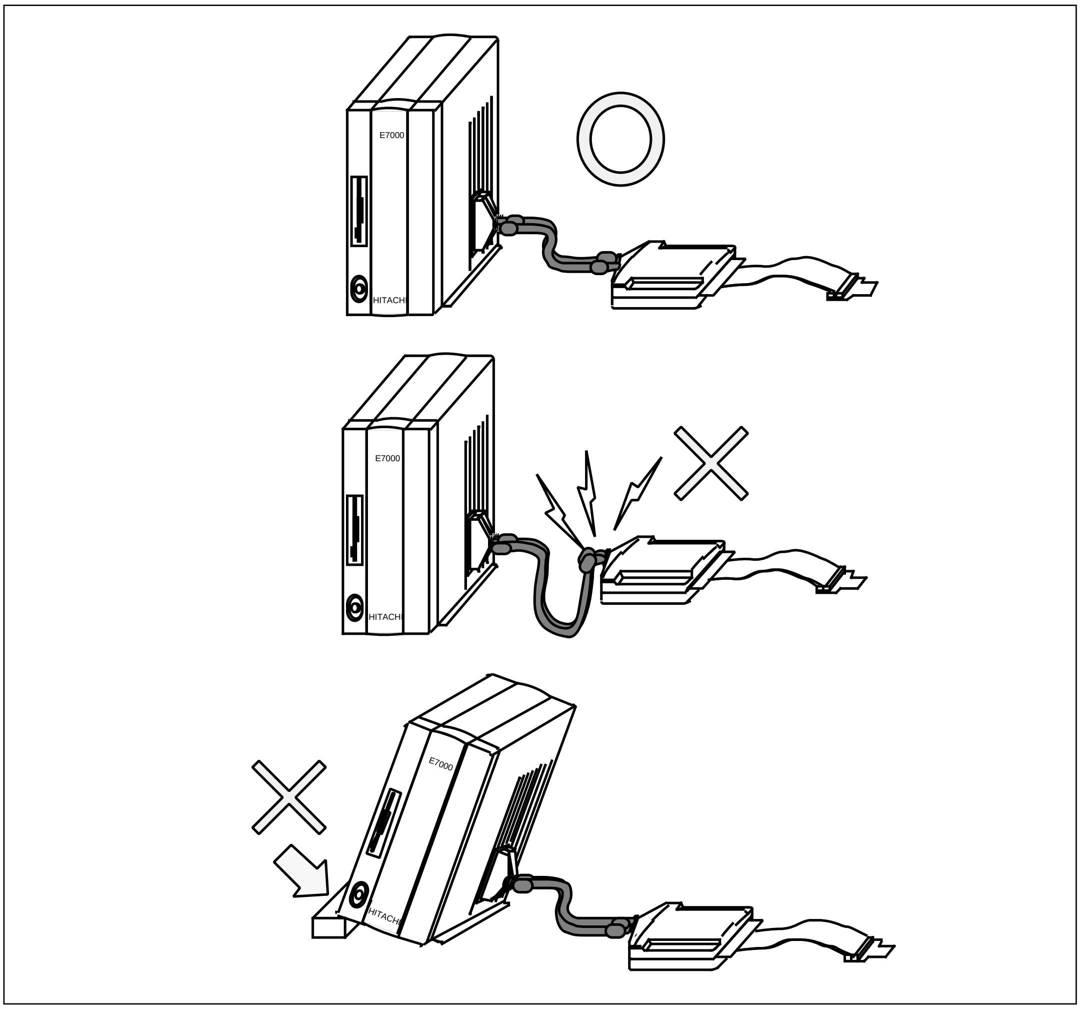


Figure C-1 Examples of Securing the Emulator Station

2. Check the power supply is off.
   Before connecting the emulator pod to the user system, check that the emulator station and the user system are powered off.

3. Connect the UVcc to user system power.
   The emulator monitors and detects the Vcc pin to determine whether or not the user system is connected. Accordingly, after connecting the user system to the emulator, be sure to supply 2.7-V to 5.5-V power to the Vcc pin. Otherwise, the emulator assumes that the user system is not connected.

Appendix D   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