User's Manual

RENESAS

SuperH™ Family E10A Emulator

Additional Document for User's Manual

SH7751 E10A HS7751KCM02HE
Renesas Microcomputer
Development Environment
System
SuperH™ Family / SH7750 Series
Specific Guide for the SH7751
E10A Emulator

Renesas Electronics
www.renesas.com

Rev.1.0 2003.12

Cautions

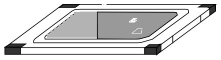
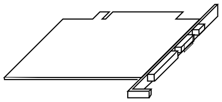
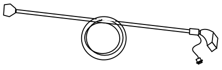
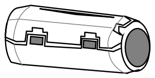
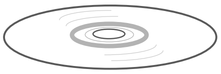
Contents

Section 1 Connecting the Emulator with the User System

1.1 Components of the Emulator

The SH7751 E10A emulator supports the SH7751. Table 1.1 lists the components of the emulator.

Table 1.1 Components of the Emulator (HS7751KCM01H, HS7751KCM02H, HS7751KCI01H, or HS7751KCI02H)

Classi-fication	Component	Appearance	Quan-tity	Remarks
Hard-ware	Card emulator	 (PCMCIA) or  (PCI)	1	HS7751KCM01H (PCMCIA: 14-pin type): Depth: 85.6 mm, Width: 54.0 mm, Height: 5.0 mm, Mass: 27.0 g HS7751KCM02H (PCMCIA: 36-pin type): Depth: 85.6 mm, Width: 54.0 mm, Height: 5.0 mm, Mass: 28.0 g HS7751KCI01H (PCI: 14-pin type): Depth: 122.0 mm, Width: 96.0 mm, Mass: 80.0 g HS7751KCI02H (PCI: 36-pin type): Depth: 122.0 mm, Width: 96.0 mm, Mass: 90.0 g
	User system interface cable		1	HS7751KCM01H (PCMCIA: 14-pin type): Length: 80 cm, Mass: 45.0 g HS7751KCM02H (PCMCIA: 36-pin type): Length: 30 cm, Mass: 55.0 g HS7751KCI01H (PCI: 14-pin type): Length: 150 cm, Mass: 86.0 g HS7751KCI02H (PCI: 36-pin type): Length: 80 cm, Mass: 69.0 g
	Ferrite core (connected with the user interface cable)		1	Countermeasure for EMI* (only for HS7751KCM02H and HS7751KCI02H)
Soft-ware	SH7751 E10A emulator setup program, SuperH™ Family E10A Emulator User's Manual, and Specific Guide for the SH7751 E10A Emulator		1	HS7751KCM01SR, HS0005KCM01HJ, HS0005KCM01HE, HS7751KCM02HJ, and HS7751KCM02HE (provided on a CD-R)

Note: The EMI is an abbreviation of the Electrical Magnetic Interference.

For EMI countermeasure, use the ferrite core by connecting the user interface cable.
When the user interface cable is connected with the emulator or user system, connect the ferrite core in the user system as shown in figure 1.1.

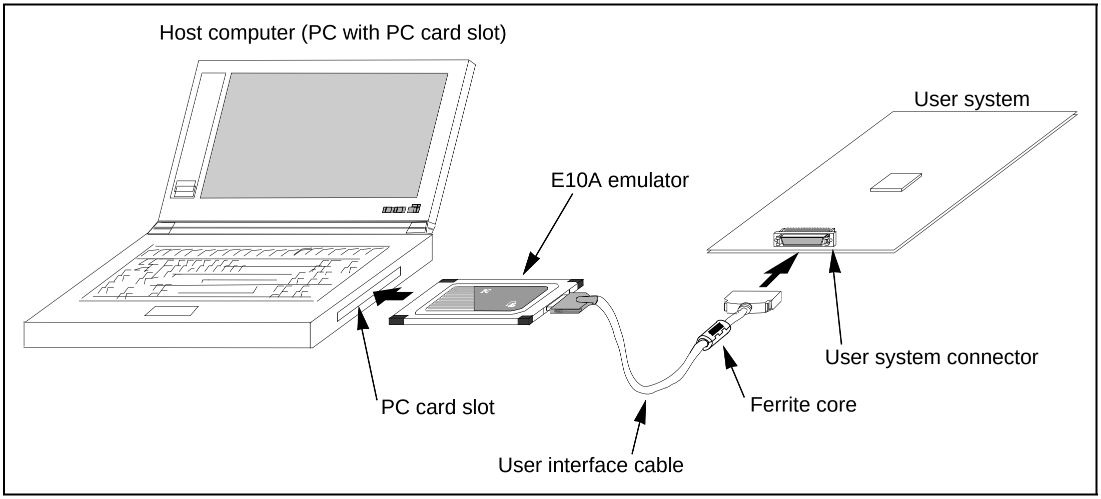


Figure 1.1   Connecting Ferrite Core

1.2 Connecting the E10A Emulator with the User System

To connect the E10A emulator (hereinafter referred to as the emulator), the H-UDI port connector must be installed on the user system to connect the user system interface cable. When designing the user system, refer to the recommended circuit between the H-UDI port connector and the MCU. In addition, read the E10A emulator user's manual and hardware manual for the related device.

Table 1.2 shows the type number of the E10A emulator, the corresponding connector type, and the use of AUD function.

Table 1.2 Type Number, AUD Function, and Connector Type

Type Number	Connector	AUD Function
HS7751KCM02H, HS7751KCI02H	36-pin connector	Available
HS7751KCM01H, HS7751KCI01H	14-pin connector	Not available

The H-UDI port connector has the 36-pin and 14-pin types as described below. Use them according to the purpose of the usage.

1. 36-pin type (with AUD function)
   The AUD trace function is supported. A large amount of trace information can be acquired in realtime. The window trace function is also supported for acquiring memory access in the specified range (memory access address or memory access data) by tracing.
2. 14-pin type (without AUD function)
   The AUD trace function cannot be used because only the H-UDI function is supported. For tracing, only the internal trace function is supported. Since the 14-pin type connector is smaller than the 36-pin type (1/2.5), the area where the connector is installed on the user system can be reduced.

1.3 Installing the H-UDI Port Connector on the User System

Table 1.3 shows the recommended H-UDI port connectors for the emulator.

Table 1.3 Recommended H-UDI Port Connectors

Connector	Type Number	Manufacturer	Specifications
36-pin connector	DX10M-36S	Hirose Electric Co., Ltd.	Screw type
	DX10M-36SE, DX10G1M-36SE		Lock-pin type
14-pin connector	2514-6002	Minnesota Mining & Manufacturing Ltd.	14-pin straight type

Note: When the 36-pin connector is used, do not connect any components under the H-UDI connector. When the 14-pin connector is used, do not install any components within 3 mm of the H-UDI port connector.

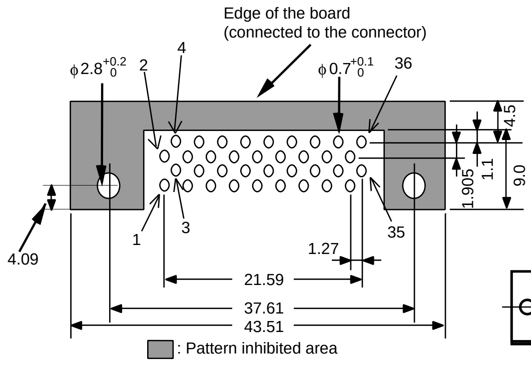
1.4 Pin Assignments of the H-UDI Port Connector

Figures 1.2 and 1.3 show the pin assignments of the 36-pin and 14-pin H-UDI port connectors, respectively.

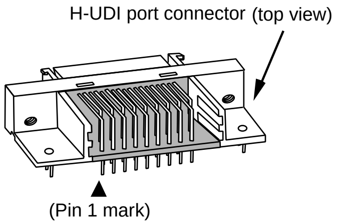
Note: Note that the pin number assignments of the H-UDI port connector shown below differ from those of the connector manufacturer.

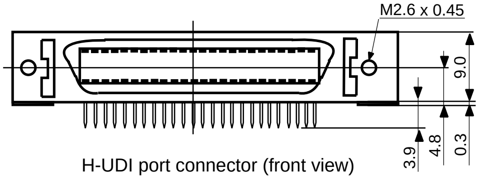
Pin No.	Signal	Input/ Output[*1]	SH7751 Pin No.	Note	Pin No.	Signal	Input/ Output[*1]	SH7751 Pin No.	Note
1	AUDCK	Output	220		19	TMS	Input	1	
2	GND	——			20	GND	——		
3	AUDATA0	Output	223		21[*2]	/TRST	Input	199	
4	GND	——			22	GND	——		
5	AUDATA1	Output	224		23	TDI	Input	5	
6	GND	——			24	GND	——		
7	AUDATA2	Output	227		25	TDO	Output	246	
8	GND	——			26	GND	——		
9	AUDATA3	Output	228		27[*2]	/ASEBRK BRKACK	I/O	245	
10	GND	——			28	GND	——		
11[*2]	AUDSYNC	Output	219		29	NC	——		
12	GND	——			30	GND	——		
13	NC	——			31[*2]	/RESET	Output	198	User reset
14	GND	——			32	GND	——		
15	NC	——			33[*3]	GND	Output		
16	GND	——			34	GND	——		
17	TCK	Input	2		35	NC	——		
18	GND	——			36	GND	——		

Notes: 1. Input to or output from the user system.
2. The slash (/) means that the signal is active-low.
3. The emulator monitors the GND signal of the user system and detects whether the user system is connected or not.



H-UDI port connector (top view)





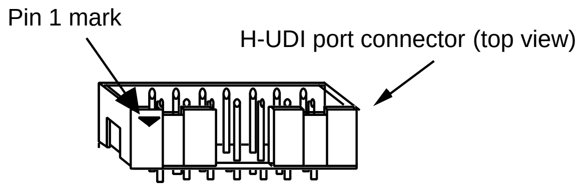
H-UDI port connector (front view)

Unit: mm

Figure 1.2 Pin Assignments of the H-UDI Port Connector (36 Pins)

Pin No.	Signal	Input/ Output*1	SH7751 Pin No.
1	TCK	Input	2
2*2	/TRST	Input	199
3	TDO	Output	246
4*2	/ASEBRK BRKACK	Input/ Output	245
5	TMS	Input	1
6	TDI	Input	5
7*2	/RESET	Output	198
11	Not connected	—	—
8 to 10 12 to 13	GND	—	—
14*3	GND	Output	—

Notes: 1. Input to or output from the user system.
2. The slash (/) means that the signal is active-low.
3. The emulator monitors the GND signal of the user system and detects whether the user system is connected or not.



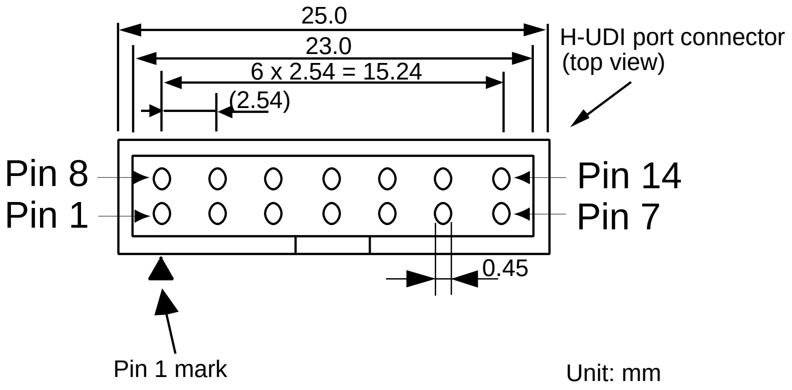


Figure 1.3 Pin Assignments of the H-UDI Port Connector (14 Pins)

1.5 Recommended Circuit for Connection between the H-UDI Port Connector and the MPU

1.5.1 Recommended Circuit for Connection (36-Pin Type)

Figure 1.4 shows a recommended circuit for connection between the H-UDI port connector (36 pins) and the MPU.

Notes: 1. Do not connect anything to the N.C. pin of the H-UDI port connector.

2. When a joined resistance is used for pull-up, it may be affected by a noise. Separate TCK from other resistances.
3. The reset signal in the user side is input to the /RESET pin (pin 198) of the SH7751. Connect this pin to the H-UDI port connector as the output from the user system.
4. The pattern between the H-UDI connector and the MPU must be as short as possible. Do not connect the signal lines to other components on the board.
5. The resistance values shown in figure 1.4 are recommended.
6. For the pin processing in cases where the emulator is not used, refer to the hardware manual of the related device.

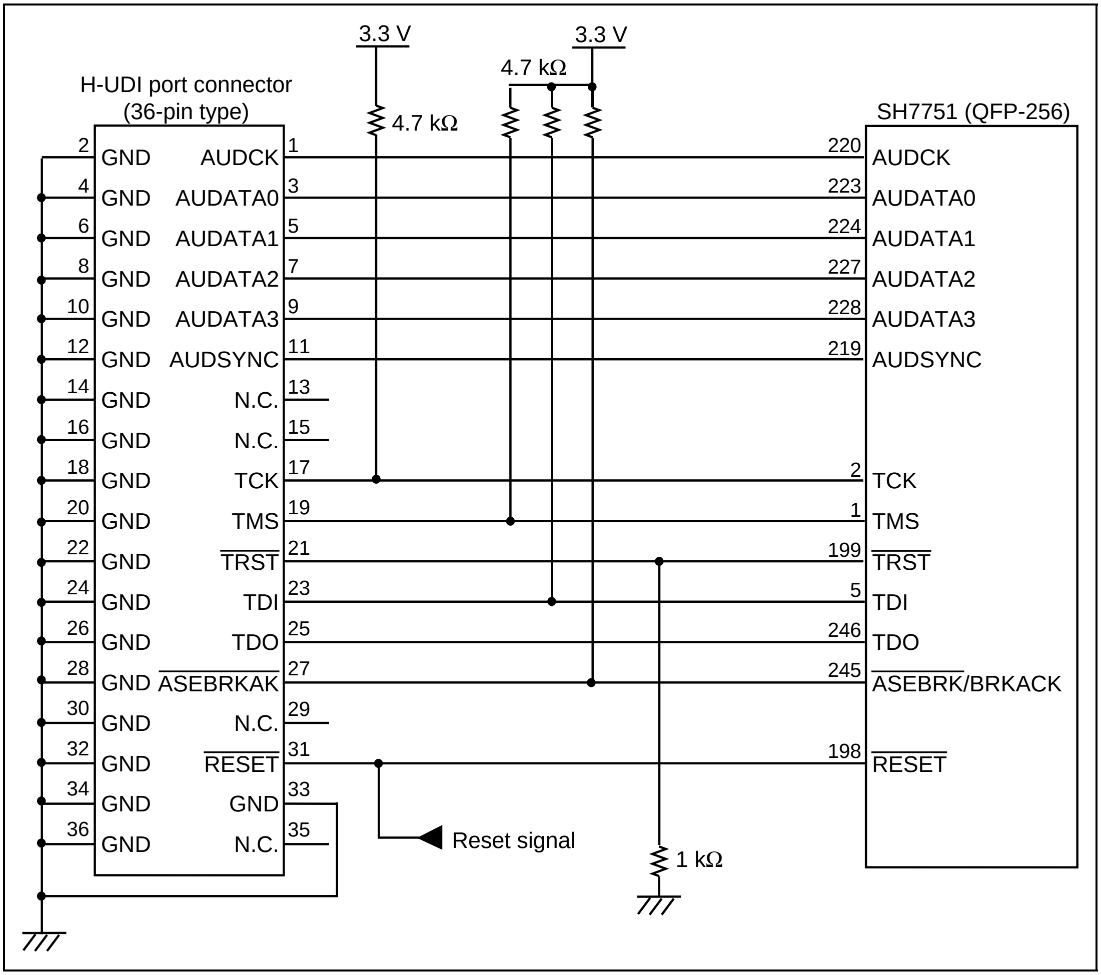


Figure 1.4 Recommended Circuit for Connection between the H-UDI Port Connector and MPU (36-Pin Type)

1.5.2 Recommended Circuit for Connection (14-Pin Type)

Figure 1.5 shows a recommended circuit for connection between the H-UDI port connector (14 pins) and the MPU.

Notes: 1. Do not connect anything to the N.C. pin of the H-UDI port connector.
2. When a joined resistance is used for pull-up, it may be affected by a noise. Separate TCK from other resistances.
3. The reset signal in the user side is input to the /RESET pin (pin 198) of the SH7751. Connect this pin to the H-UDI port connector as the output from the user system.
4. The pattern between the H-UDI connector and the MPU must be as short as possible. Do not connect the signal lines to other components on the board.
5. The resistance values shown in figure 1.5 are recommended.
6. For the pin processing in cases where the emulator is not used, refer to the hardware manual of the related device.

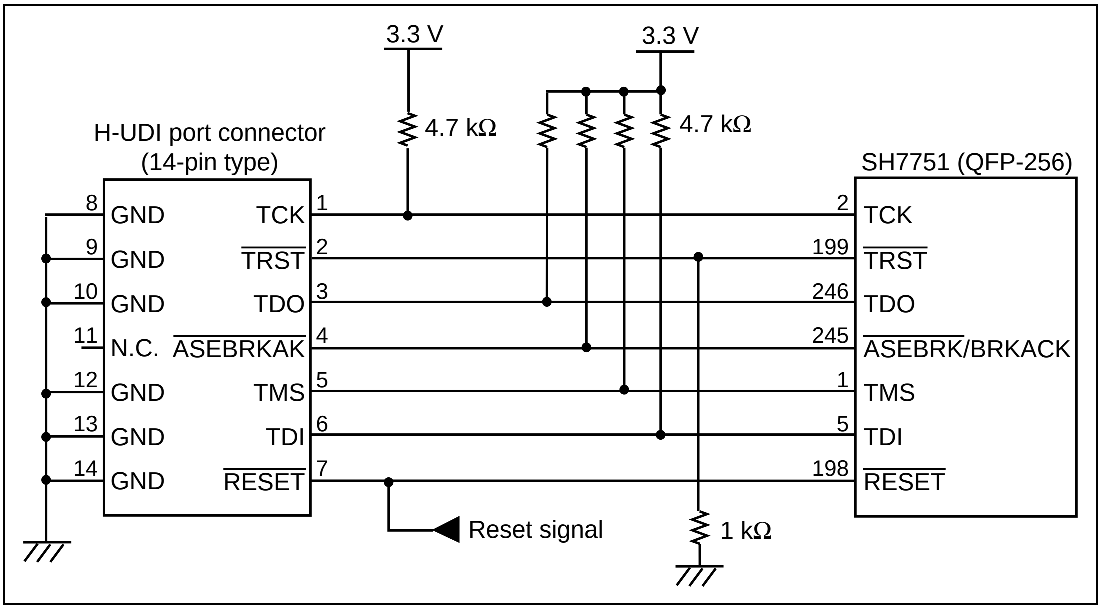


Figure 1.5 Recommended Circuit for Connection between the H-UDI Port Connector and MPU (14-Pin Type)

Section 2 Specifications of the SH7751 E10A Emulator's Software

2.1 Differences between the SH7751 and the Emulator

1. When the emulator system is initiated, it initializes the general registers and part of the control registers as shown in table 2.1.

Table 2.1 Register Initial Values at Emulator Power-On

Register	Emulator at Power-on
R0 to R14	H'00000000
R15 (SP)	H'00000000
R0_BANK to R7_BANK	H'00000000
PC	H'A0000000
SR	H'700000F0
GBR	H'00000000
VBR	H'00000000
MACH	H'00000000
MACL	H'00000000
PR	H'00000000
DBR	H'00000000
SGR	H'00000000
SPC	H'00000000
SSR	H'000000F0
FPUL	H'00000000
FPSCR	H'00040001
FR0 to FR15	H'00000000
XF0 to XF15	H'00000000

2. The emulator uses the H-UDI; do not access the H-UDI.

3. Low-Power States (Sleep, Standby, and Module Standby)

   For low-power consumption, the SH7751 has sleep, standby, and module standby modes.

   The sleep and standby modes are switched using the SLEEP instruction. When the emulator is used, the sleep and standby modes can be cleared by either normal clearing or with the [Stop] button. Note that, however, if a command has been entered in standby mode or module standby mode, the TIMEOUT error is displayed.

Notes: 1. After the sleep mode is cleared by a break, execution restarts at the instruction following the SLEEP instruction.

2. If the memory is accessed or modified in sleep mode, the sleep mode is cleared and execution starts at the instruction following the SLEEP instruction.

3. If an operation such as the command input is performed in the hardware standby state, the command from the emulator cannot be used, or the state cannot be cancelled by the [Stop] button.

4. When the SLEEP instruction is executed by STEP-type commands, set [Rate] to 6 to use [Step…] from the [Run] menu. If [Rate] is 5 or less, a COMMUNICATION TIMEOUT error occurs.

4. Reset Signals

   The SH7751 reset signals are only valid during emulation started with clicking the GO or STEP-type button. If these signals are input from the user system in command input wait state, they are not sent to the SH7751.

Note: Do not break the user program when pins /RESET, /MRESET, /BREQ, and /RDY are being low. A TIMEOUT error will occur. If the /BREQ and /RDY pins are fixed to low during break, a TIMEOUT error will occur at memory access.

5. Direct Memory Access Controller (DMAC)

   The DMAC operates even when the emulator is used. When a data transfer request is generated, the DMAC executes DMA transfer.

6. Memory Access during User Program Execution

   When a memory is accessed from the memory window, etc. during user program execution, the user program is resumed after it has stopped in the E10A emulator to access the memory. Therefore, realtime emulation cannot be performed.

   The stopping time of the user program is as follows:

   Environment:

   Host computer: 800 MHz (Pentium® III)
   OS: Windows® 2000
   SH7751: 50 MHz (CPU clock)
   JTAG clock: 15 MHz

   When a one-byte memory is read from the command-line window, the stopping time will be about 35 ms.

7. Interrupt

   When the NMIB bit in the ICR register is 1, the NMI interrupt is accepted during break and the program is executed from the NMI interrupt vector. If the program cannot return normally from the NMI interrupt routine or the value in the general-purpose register is not guaranteed, a communication timeout error will occur.

8. Memory Access during User Program Break

   The emulator can download the program for the flash memory area (refer to section 6.22, Download Function to the Flash Memory Area, in the SuperH™ Family E10A Emulator User's Manual). Other memory write operations are enabled for the RAM area. Therefore, an operation such as memory write or BREAKPOINT should be set only for the RAM area. When the memory area can be written by the MMU, do not perform memory write, BREAKPOINT break, or downloading.

9. Cache Operation during User Program Break

   When cache is enabled, the emulator accesses the memory by the following methods:

   - At memory write: Writes through the cache, then writes to the memory.
   - At memory read: Does not change the cache write mode that has been set.

   Therefore, when memory read or write is performed during user program break, the cache state will be changed.

10. UBC

    When [User] is specified in the [UBC mode] list box in the [Configuration] dialog box, the UBC can be used in the user program.
    Do not use the UBC in the user program as it is used by the E10A emulator when [EML] is specified in the [UBC mode] list box in the [Configuration] dialog box.

11. MFI Boot Mode

    When the MFI boot mode is used, be sure to allocate the boot program from the top of MFRAM.

12. Using RWDT

    At power-on reset, the operation of RWDT is enabled. When RWDT is not used, be sure to disable the operation of RWDT at the top of the user-reset program.

13. Memory Access during Break

    In the enabled MMU, when a memory is accessed and a TLB error occurs during break, it can be selected whether the TLB exception is controlled or the program jumps to the user exception handler in [TLB Mode] in the [Configuration] dialog box. When [TLB miss exception is enable] is selected, a Communication Timeout error will occur if the TLB exception handler does not operate correctly. When [TLB miss exception is disable] is selected, the program does not jump to the TLB exception handler even if a TLB exception occurs. Therefore, if the TLB exception handler does not operate correctly, a Communication Timeout error will not occur but the memory contents may not be correctly displayed.

14. Loading Sessions

    Information in [JTAG clock] of the [Configuration] dialog box cannot be recovered by loading sessions. Thus the TCK value will be as follows:

    - When HS7751KCI01H or HS7751KCI02H is used: TCK = 1.031 MHz
    - When HS7751KCM01H or HS7751KCM02H is used: TCK = 0.937 MHz

15. [IO] Window

    - Display and modification

      Do not change values of the User Break Controller because it is used by the emulator.

      For each watchdog timer register, there are two registers to be separately used for write and read operations.

Table 2.2 Watchdog Timer Register

Register Name	Usage	Register
WTCSR(W)	Write	Watchdog timer control/status register
WTCNT(W)	Write	Watchdog timer counter
WTCSR(R)	Read	Watchdog timer control/status register
WTCNT(R)	Read	Watchdog timer counter

- The watchdog timer operates only when the user program is executed. Do not change the value of the frequency change register in the [IO] window or [Memory] window.
- The internal I/O registers can be accessed from the [IO] window. However, note the following when accessing the SDMR register of the bus-state controller. Before accessing the SDMR register, specify addresses to be accessed in the I/O-register definition file (SH7751.IO) and then activate the HEW. After the I/O-register file is created, the MPU's specification may be changed. If each I/O register in the I/O-register definition file differs from addresses described in the hardware manual, change the I/O-register definition file according to the description in the hardware manual. The I/O-register definition file can be customized depending on its format. Note that, however, the E10A emulator does not support the bit-field function.

- Verify
  In the [IO] window, the verify function of the input value is disabled.

16. Illegal Instructions
    If illegal instructions are executed by STEP-type commands, the emulator cannot go to the next program counter.

2.2 Specific Functions for the SH7751 E10A Emulator

2.2.1 Emulator Driver Selection

Table 2.3 shows drivers which are selected in the [E10A Driver Details] dialog box.

Table 2.3 Type Number and Driver

Type Number	Driver
HS7751KCM01H	E10A PC Card Driver 3
HS7751KCI01H	E10A PCI Card Driver 3
HS7751KCM02H	E10A PC Card Driver 4
HS7751KCI02H	E10A PCI Card Driver 4

2.2.2 Break Condition Functions

In addition to BREAKPOINT functions, the emulator has Break Condition functions. Eight types of conditions can be set (Break Condition 1,2,3,4,5,6,7,8). Break Condition 5,6 use the user break controller (UBC). Table 2.4 lists these conditions of Break Condition.

Table 2.4 Types of Break Conditions

Break Condition Type	Description
Address bus condition (Address)	Breaks when the SH7751 address bus value or the program counter value matches the specified value.
Data bus condition (Data)	Breaks when the SH7751 data bus value matches the specified value. Byte, word, or longword can be specified as the access data size.
ASID condition (ASID)	Breaks when the SH7751 ASID value matches the specified condition.
Bus state condition (Bus State)	There are two bus state condition settings: Read/write condition: Breaks at the read or write cycle. Bus state condition: Breaks when the operating state in an SH7751 bus cycle matches the specified condition.
LDTLB instruction break condition	Breaks when the SH7751 executes the LDTLB instruction.
Internal I/O break condition	Breaks when the SH7751 accesses the internal I/O.

Note: For details on window function and command-line syntax, refer to the online help.

Table 2.5 lists the combinations of conditions that can be set under Break Condition 1, 2, 3, 4, 5, 6, 7, 8.

Table 2.5 Dialog Boxes for Setting Break Conditions

	Dialog Box			
	[Break Condition 1,5] Dialog Box	**[Break Condition 2,3, 4,6] Dialog Box**	**[Break Condition 7] Dialog Box**	**[Break Condition 8] Dialog Box**
Address bus condition (Address)	O	O	X	X
Data bus condition (Data)	O	X	X	X
ASID condition (ASID)	O	O	X	X
Read/write specification	O	O	X	X
Data access	O	O	X	X
Before/after execution	O	O	X	X
Sequential break	O	O	X	X
LDTLB instruction break	X	X	X	O
Internal I/O break	X	X	O	X

Note: O: Can be set in the dialog box.
X: Cannot be set in the dialog box.

The SH7751 E10A emulator has sequential break functions. Table 2.6 lists the sequential break conditions.

Table 2.6 Sequential Break Conditions

No.	Break Condition	Description
1	Sequential break condition 2-1	Program is halted when Break Condition 2 and Break Condition 1 are satisfied in that order. Break Condition 2,1 should be set.
2	Sequential break condition 3-2-1	Program is halted when Break Condition 3, Break Condition 2, and Break Condition 1 are satisfied in that order. Break Condition 3,2,1 should be set.
3	Sequential break condition 4-3-2-1	Program is halted when Break Condition 4, Break Condition 3, Break Condition 2, and Break Condition 1 are satisfied in that order. Break Condition 4,3,2,1 should be set.
4	Sequential break condition 6-5	Program is halted when Break Condition 6 and Break Condition 5 are satisfied in that order. Break Condition 6,5 should be set.

Note: Sequential breaks can be specified by the [Configuration] dialog box.
Numbers 1 to 3 in table 2.6 can be set in the [Emulation_mode] list box in the [Configuration] dialog box or with the Go_option command. For details on command-line syntax, refer to the online help function.
Number 4 can be set in the [UBC_mode] list box in the [Configuration] dialog box or with the UBC_mode command. For details on command-line syntax, refer to the online help.

2.2.3 Trace Functions

The SH7751 E10A emulator supports the trace functions listed in table 2.7.

Table 2.7 Trace Functions

Function	Internal Trace	AUD Trace
Branch trace	Supported (eight branches) (32 branches at continuous trace)	Supported
Internal I/O access trace	Supported (non realtime)	Not supported
LDTLB instruction execution trace	Supported (non realtime)	Not supported
Range memory access trace	Not supported	Supported
Software trace	Not supported	Supported

Table 2.8 shows the type numbers that the AUD function can be used.

Table 2.8 Type Number and AUD Function

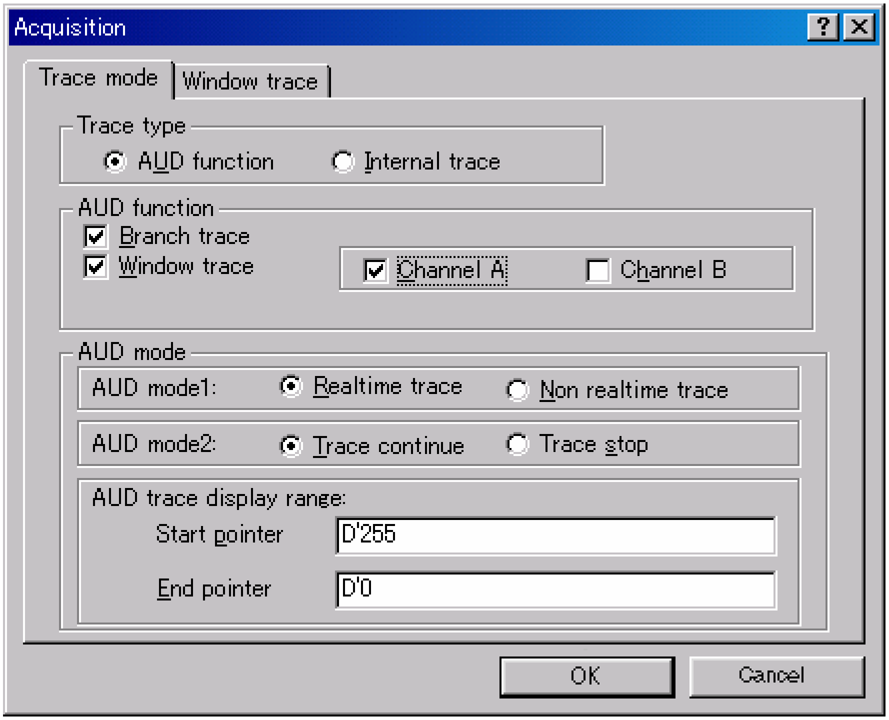
Type Number	AUD Function
HS7751KCM01H, HS7751KCI01H	Not supported
HS7751KCM02H, HS7751KCI02H	Supported

AUD Trace Functions: This function is operational when the AUD pin of the device is connected to the emulator. Table 2.9 shows the AUD trace acquisition mode that can be set in each trace function.

Table 2.9 AUD Trace Acquisition Mode

Type	Mode	Description
Continuous trace occurs	Realtime trace	When the next branch occurs while the trace information is being output, the trace information being output is output but the next trace information is not output. The user program can be executed in realtime, but some trace information may be lost.
	Non realtime trace	When the next branch occurs while the trace information is being output, the CPU stops operations until the information is output. The user program is not executed in realtime.
Trace buffer full	Trace continue	This function overwrites the oldest trace information to store the latest trace information.
	Trace stop	After the trace buffer becomes full, the trace information is no longer acquired. (The user program is continuously executed.)

To set the AUD trace acquisition mode, click the [Trace] window with the right mouse button and select [Setting] from the pop-up menu to display the [Acquisition] dialog box. The AUD trace acquisition mode can be set in the [AUD mode1] or [AUD mode2] group box in the [Trace mode] page of the [Acquisition] dialog box.


Acquisition

Trace mode | Window trace

Trace type: [✓ AUD function] Internal trace

AUD function:
- [✓] Branch trace
- [✓] Window trace
- [✓] Channel A
- [ ] Channel B

AUD mode:
- AUD mode1: [✓ Realtime trace] Non realtime trace
- AUD mode2: [✓ Trace continue] Trace stop

AUD trace display range:
- Start pointer: D'255
- End pointer: D'0

Figure 2.1   [Trace mode] Page

When the AUD trace function is used, select the [AUD function] radio button in the [Trace type] group box of the [Trace mode] page.

(a) Branch Trace Function

The branch source and destination addresses and their source lines are displayed.

Branch trace can be acquired by selecting the [Branch trace] check box in the [AUD function] group box of the [Trace mode] page.

(b) Window Trace Function

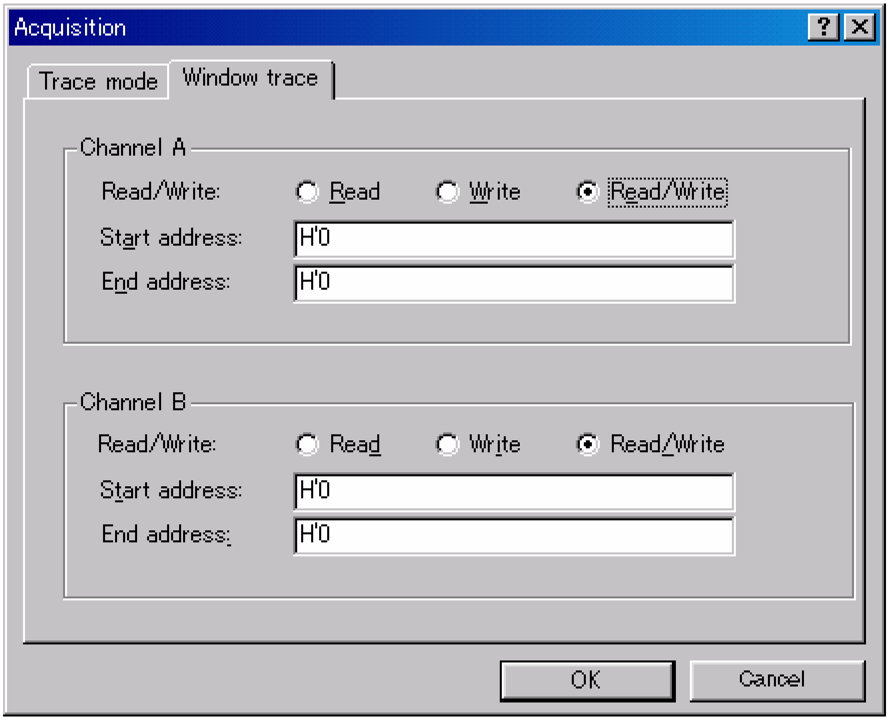
Memory access in the specified range can be acquired by trace.

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

[Setting Method]

(i) Select the [Channel A] and [Channel B] check boxes in the [AUD function] group box of the [Trace mode] page. Each channel will become valid.

(ii) Open the [Window trace] page and specify the bus cycle and memory range that are to be set for each channel.



Channel A

Read/Write: Read Write [✓ Read/Write]

Start address: H'0

End address: H'0

Channel B

Read/Write: Read Write [✓ Read/Write]

Start address: H'0

End address: H'0

Figure 2.2 [Window trace] Page

(c) Software Trace Function

Note: This function can be supported with SHC compiler (manufactured by Renesas Technology Corp.; including OEM and bundle products) V6.0 or later.

When a specific instruction is executed, the PC value at execution and the contents of one general register are acquired by trace. Describe the Trace(x) function (x is a variable name) to be compiled and linked beforehand. For details, refer to the SHC manual.

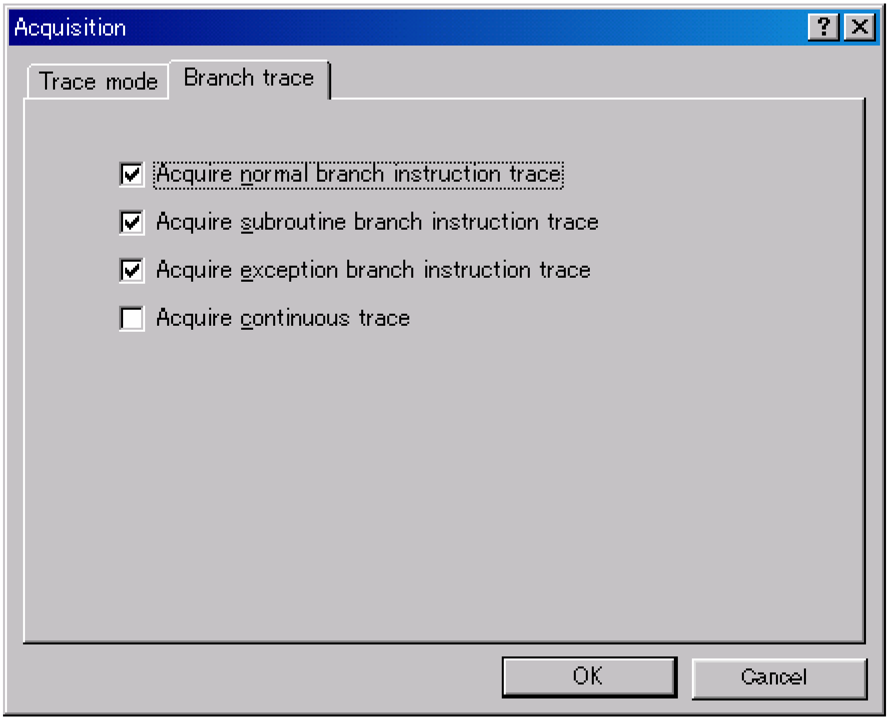
When the load module is loaded on the emulator and a valid software trace function is executed, the PC value that has executed the Trace(x) function, the general register value for x, and the source lines are displayed.

To activate the software trace function, select the [Software trace] check box in the [AUD function] group box of the [Trace mode] page.

Notes on AUD Trace:

1. If a TLB error occurs in the trace acquisition information display, the [Trace] window displays the contents. However, mnemonics, operands, or source is not displayed.
2. When the trace display is performed during user program execution, the mnemonics, operands, or source is not displayed.
3. When MMU settings are modified or when a user program is changed after GO command completion before trace display, the displayed mnemonics, operands, or source may not be correct.
4. The AUD trace function outputs the differences between newly output branch source addresses and previously output branch source addresses. The window trace function outputs the differences between newly output addresses and previously output addresses. If the previous branch source address is the same as the upper 16 bits, the lower 16 bits are output. If it matches the upper 24 bits, the lower 8 bits are output. If it matches the upper 28 bits, the lower 4 bits are output.
   The emulator regenerates the 32-bit address from these differences and displays it in the [Trace] window. If the emulator cannot display the 32-bit address, it displays the difference from the previously displayed 32-bit address.
5. If the 32-bit address cannot be displayed, the source line is not displayed.
6. In the SH7751 E10A emulator, when multiple loops are performed to reduce the number of AUD trace displays, only the IP counts up.
7. In the SH7751 E10A emulator, the maximum number of trace display pointers is as follows:
   When HS7751KCM02H is used: D'8191 to -0
   When HS7751KCI02H is used: D'32767 to -0
   However, the maximum number of trace display pointers differs according to the AUD trace information to be output. Therefore, the above pointers cannot be always acquired.

Internal Trace Function: This function is activated by selecting the [Internal trace] radio button in the [Trace type] group box of the [Trace mode] page. See figure 2.1, [Trace mode] Page. The internal trace functions are also activated by selecting each check box on the [Branch trace] page.



[✓] Acquire normal branch instruction trace
[✓] Acquire subroutine branch instruction trace
[✓] Acquire exception branch instruction trace
[ ] Acquire continuous trace

Figure 2.3 [Branch trace] Page

Table 2.10 shows the internal trace functions.

Table 2.10 Internal Trace Functions

Function	Description
Branch instruction trace	Traces and displays the branch instructions. The branch source address and branch destination address for the eight latest branch instructions are displayed. There are three kinds of branch instruction trace: • Normal branch instruction trace Traces and displays the normal branch instructions. The normal branch instructions are the BF, BF/S, BT/S, BRA, BRAF, and JMP instructions. To use this function, select the [Acquire normal branch instruction trace] check box in the [Branch trace] page. • Subroutine branch instruction trace Traces and displays the subroutine branch instructions. The subroutine branch instructions are the BSR, BSRF, JSR, and RTS instructions. To use this function, select the [Acquire subroutine branch instruction trace] check box in the [Branch trace] page. • Exception branch instruction trace Traces and displays the exception branch instruction. The exception branch instruction is the RTE instruction. In addition, all the exception and interrupt operations are traced. To use this function, select the [Acquire exception branch instruction trace] check box in the [Branch trace] page.
Continuous trace	Acquires the trace information continuously. This is called continuous trace. For the branch instruction trace, eight-branch information can be repeatedly acquired a maximum of four times. Select the [Acquire continuous trace] check box in the [Branch trace] page. If continuous trace is selected, realtime trace cannot be performed.

Table 2.10 Internal Trace Functions (cont)

Function	Description
Internal I/O trace	Traces and displays the address and data that access the internal I/O area. To use this function, select the [Get trace information of internal I/O Area] radio button in the [Break Condition 7] dialog box and the [Acquire continuous trace] check box in the [Branch trace] page.
LDTLB instruction execution trace	Traces and displays the address that executes the LDTLB instruction. To use this function, select the [Get trace information of LDTLB instruction] radio button in the [Break Condition 8] dialog box and the [Acquire continuous trace] check box in the [Branch trace] page.

Notes: 1. When the continuous trace is not used, trace acquisition of the eight latest branch instructions is enabled.

2. If an interrupt is generated at the program execution start or end, including a step execution, the emulator address may be acquired. In such a case, the following message will be displayed. Ignore this address because it is not a user program address.

   *** EML ***

3. If a completion-type exception occurs during exception branch acquisition, the next address to the address in which an exception occurs is acquired.

4. When a user interrupt is enabled by the INTERRUPT command during the emulator command wait state or user program execution, an interrupt that is generated at the program execution start or end, including a step execution, can be traced in realtime.

5. When the [Acquire continuous trace] check box is selected, do not perform memory access during emulation.

6. When internal I/O trace or LDTLB instruction trace is performed, select the [Acquire continuous trace] check box.

7. When the [Acquire continuous trace] check box is selected, 32 trace information data can be acquired. In this case, however, since the user program stops at constant intervals, the processing speed is decreased compared with the case where the [Acquire continuous trace] check box is not selected.

8. Trace information cannot be acquired for the following branch instructions:
   - The BF and BT instructions whose displacement value is 0
   - Branch to H'A0000000 by reset

9. When the [Acquire continuous trace] check box is selected, and when either the [Get trace information of internal I/O area] radio button (internal I/O trace enabled) or the [Get trace information of LDTLB instruction] radio button is selected (LDTLB instruction trace enabled) with the [Break Condition 5] dialog box,
    - An internal I/O trace cannot be made with the Step In function.
    - The LDTLB instruction and internal I/O trace cannot be performed with the Step Over function.
10. When continuous trace is used, do not enable user interrupt by the INTERRUPT command during the emulator command wait state or user program execution.

2.2.4 Notes on Using the JTAG Clock (TCK) and AUD Clock (AUDCK)

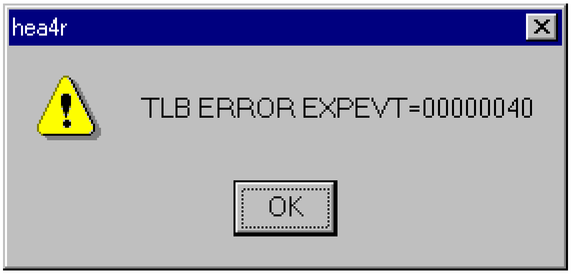
1. Set the JTAG clock (TCK) frequency to lower than the frequency of the SH7751 peripheral module clock (CKP).
2. Set the AUD clock (AUDCK) frequency to 50 MHz or below for PCMCIA and PCI cards.

2.2.5 Notes on Setting the [Breakpoint] Dialog Box

1. When an odd address is set, the next lowest even address is used.
2. A BREAKPOINT is accomplished by replacing instructions of the specified address. Accordingly, it can be set only to the internal RAM area. However, a BREAKPOINT cannot be set to the following addresses:
    - An address whose memory content is H'003B
    - An area other than the internal RAM
    - An instruction in which Break Condition 3 is satisfied
    - A slot instruction of a delayed branch instruction

   In addition, do not perform memory write, BREAKPOINT, or download even if the memory space can only be written by the MMU.
3. During step execution, a BREAKPOINT is disabled.
4. Conditions set at Break Condition 3 are disabled when an instruction to which a BREAKPOINT has been set is executed. Do not set a BREAKPOINT to an instruction in which Break Condition 3 is satisfied.
5. When execution resumes from the address where a BREAKPOINT is specified, single-step execution is performed at the address before execution resumes. Therefore, realtime operation cannot be performed.
6. When a BREAKPOINT is set to the slot instruction of a delayed branch instruction, the PC value becomes an illegal value. Accordingly, do not set a BREAKPOINT to the slot instruction of a delayed branch instruction.

7. When the [Normal] option is selected in the [Memory area] group box in the [General] page of the [Configuration] dialog, a BREAKPOINT is set to a physical address or a virtual address according to the SH7751 MMU status during command input when the VPMAP_SET command setting is disabled. The ASID value of the SH7751 PTEH register during command input is used. When VPMAP_SET command setting is enabled, a BREAKPOINT is set to a physical address into which address translation is made according to the VP_MAP table. However, for addresses out of the range of the VP_MAP table, the address to which a BREAKPOINT is set depends on the SH7751 MMU status during command input. Even when the VP_MAP table is modified after BREAKPOINT setting, the address translated at BREAKPOINT setting is valid.
8. When the [Physical] option is selected in the [Memory area] group box in the [General] page of the [Configuration] dialog box, a BREAKPOINT is set to a physical address. A BREAKPOINT is set after disabling the SH7751 MMU during program execution. After setting, the MMU is returned to the original state. When a break occurs at the corresponding virtual address, the cause of termination displayed in the status bar and the [Status] window is ILLEGAL INSTRUCTION, not BREAKPOINT.
9. When the [Virtual] option is selected in the [Memory area] group box in the [General] page of the [Configuration] dialog box, a BREAKPOINT is set to a virtual address. A BREAKPOINT is set after enabling the SH7751 MMU during program execution. After setting, the MMU is returned to the original state. When an ASID value is specified, the BREAKPOINT is set to the virtual address corresponding to the ASID value. The emulator sets the BREAKPOINT after rewriting the ASID value to the specified value, and returns the ASID value to its original value after setting. When no ASID value is specified, the BREAKPOINT is set to a virtual address corresponding to the ASID value at command input.
10. If a TLB error occurs during virtual address setting, the following message box will be displayed.



TLB ERROR EXPEVT=00000040

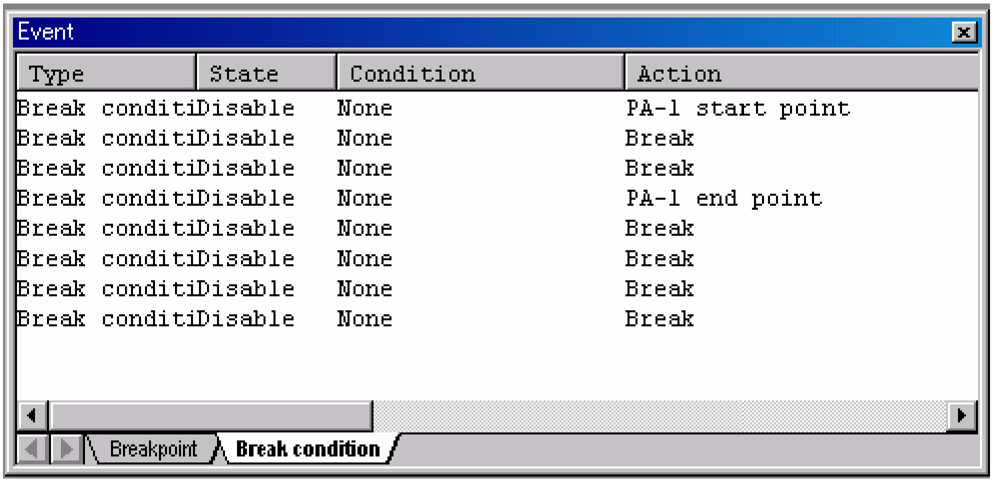
Figure 2.4 Message Box for Clearing a TLB Error

If a program is executed again without clearing the BREAKPOINT set at the address in which the TLB error occurs, a TLB error will occur again. Accordingly, clear the BREAKPOINT before execution.

11. An address (physical address) to which a BREAKPOINT is set is determined when the BREAKPOINT is set. Accordingly, even if the VP_MAP table is modified after BREAKPOINT setting, the BREAKPOINT address remains unchanged. When a BREAKPOINT is satisfied with the modified address in the VP_MAP table, the cause of termination displayed in the status bar and the [Status] window is ILLEGAL INSTRUCTION, not BREAKPOINT.
12. When a BREAKPOINT is set to the cacheable area, the cache block containing the BREAKPOINT address is filled immediately before and after user program execution.
13. While a BREAKPOINT is set, the contents of the instruction cache are disabled at execution completion.

2.2.6 Notes on Setting the [Break Condition] Dialog Box and the BREAKCONDITION_ SET Command

1. When [Go to cursor], [Step In], [Step Over], or [Step Out] is selected, the settings of Break Condition 3 are disabled.
2. Break Condition 3 is disabled when an instruction to which a BREAKPOINT has been set is executed. Accordingly, do not set a BREAKPOINT to an instruction which satisfies Break Condition 3.
3. When a Break Condition is satisfied, emulation may stop after two or more instructions have been executed.
4. If a PC break before execution is set to the slot instruction after a delayed branch instruction, user program execution cannot be terminated before the slot instruction execution; execution stops before the branch destination instruction.
5. Break Condition 5,6 use the UBC. When the UBC is used in the user program, change the UBC setting for users by using the [UBC_mode] list box in the [Configuration] dialog box or the UBC_mode command.
6. Break Condition 1,4 are used as the measurement range in the performance measurement function when [PA-1 start point] and [PA-1 end point] are displayed on the [Action] part in the [Break condition] sheet of the [Event] window. For setting the performance measurement function, refer to section 2.2.8, Performance Measurement Functions. This applies when the Break Condition is displayed with the BREAKCONDITION_DISPLAY command in the command-line function. In this case, a break does not occur when Break Condition 1,4 are satisfied.



Type	State	Condition	Action
Break conditi	Disable	None	PA-1 start point
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break
Break conditi	Disable	None	PA-1 end point
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break

Figure 2.5 [Event] Window

2.2.7 Notes on Setting the UBC_MODE Command

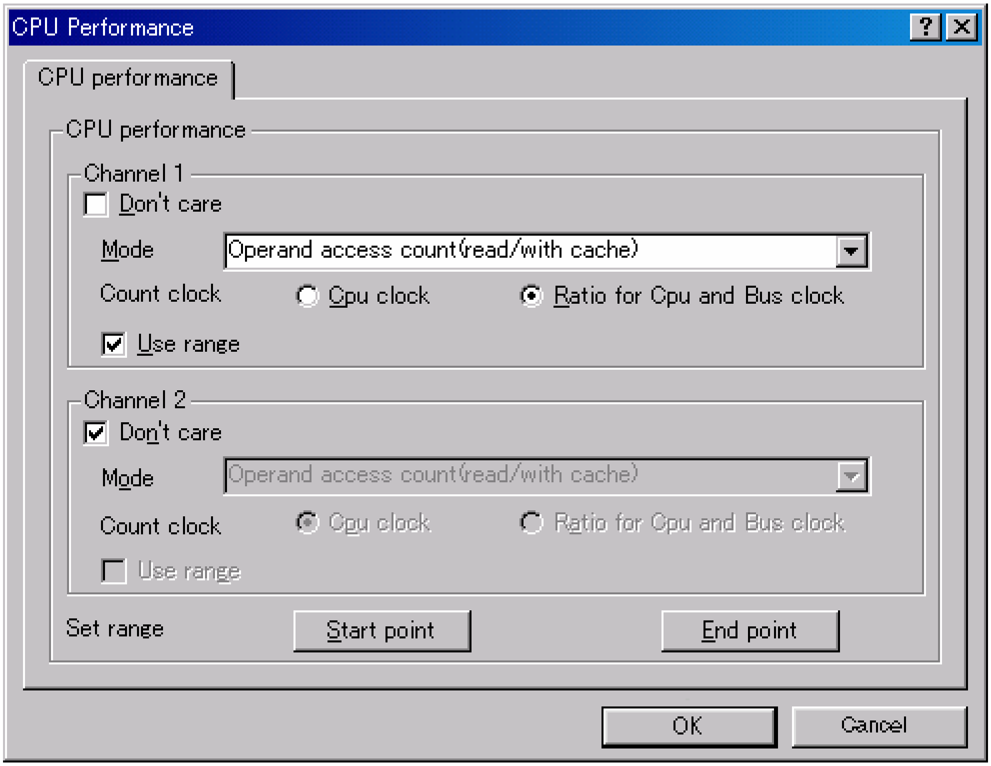
In the [Configuration] window, if [User] is set while the [UBC mode] list box has been set, the STEP-type commands that use Break Condition 2 for implementation cannot be used.

2.2.8 Performance Measurement Function

The SH7751 E10A emulator supports the performance measurement function.

1. Setting the performance measurement conditions

   To set the performance measurement conditions, use the [CPU Performance] dialog box and the PERFORMANCE_SET command. When any line on the [Performance Analysis] window is clicked with the right mouse button, the popup menu is displayed and the [CPU Performance] dialog box is displayed by selecting [Setting].


CPU Performance

CPU performance

CPU performance

Channel 1

☐ Don't care

Mode: Operand access count(read/with cache)

Count clock: ○ Cpu clock [✓ ● Ratio for Cpu and Bus clock]

☑ Use range

Channel 2

☑ Don't care

Mode: Operand access count(read/with cache)

Count clock: [✓ ● Cpu clock] ○ Ratio for Cpu and Bus clock

☐ Use range

Set range: Start point | End point

OK | Cancel

Figure 2.6 [CPU Performance] Dialog Box

Note: For the command line syntax, refer to the online help.

The emulator measures how many times the conditions of the user program specified with the performance analysis function are satisfied. For this function, two events can be measured simultaneously and the following conditions can be specified:

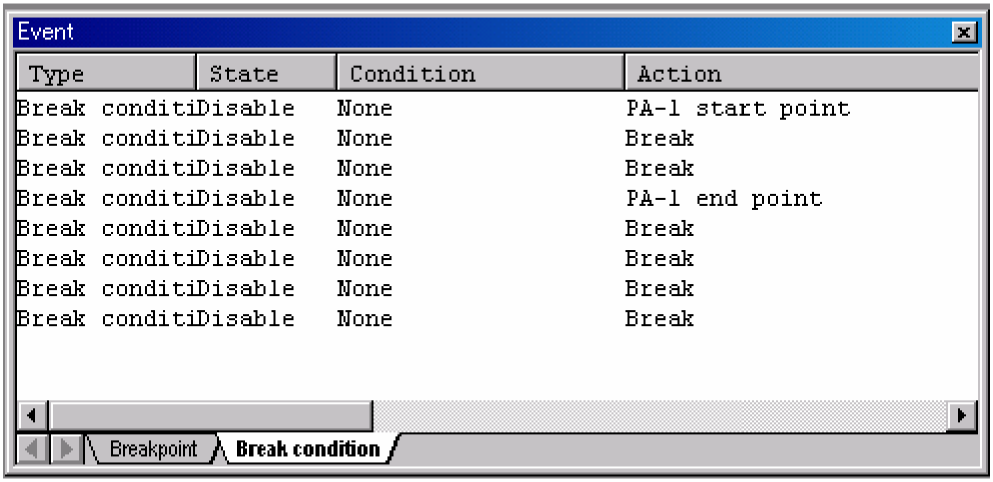
(a) Measurement range

One of the following ranges can be specified by either of measurement channels 1 and 2.

1. From the start to the end of the user program execution
2. From the satisfaction of the condition set in Break Condition 1 to the satisfaction of the condition set in Break Condition 4

When the first range is specified, the measurement result includes a several-cycle error for one user program execution. Therefore, do not specify this range when the step is to be executed. In addition, the user program execution stops when continuous trace is used; again, do not specify the first range in this case.

In the second range, [PA-1 start point] and [PA-1 end point] are displayed on the [Action] part in the [Break condition] sheet of the [Event] window.

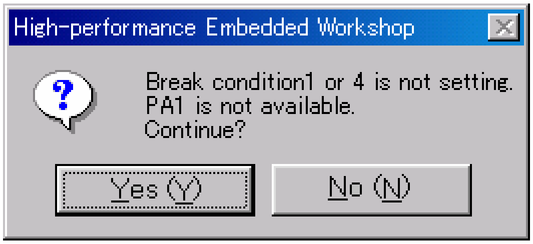


Type	State	Condition	Action
Break conditi	Disable	None	PA-1 start point
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break
Break conditi	Disable	None	PA-1 end point
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break
Break conditi	Disable	None	Break

Figure 2.7 [Event] Window ([Break condition] Sheet)

In this case, break will not occur when the conditions of Break Condition 1 and Break Condition 4 are satisfied.

Note: When the range is specified, be sure to set the measurement start and end conditions for Break Condition 1 and Break Condition 4, respectively, and then execute the user program. If Break Condition 1 or Break Condition 4 is not set and the user program is executed, performance is not measured normally. In this case, the following dialog box is displayed.



Break condition1 or 4 is not setting.
PA1 is not available.
Continue?

Figure 2.8 [High-performance Embedded Workshop] Dialog Box

(b) Measurement item

Items are measured with [Channel 1 to 2] in the [CPU Performance] dialog box. Maximum two conditions can be specified at the same time. Table 2.11 shows the measurement items (Options in table 2.11 are parameters for <mode> of the PERFORMANCE_SET command. They are displayed for NAME in the [Performance Analysis] window).

Table 2.11 Measurement Items

Event	Keyword	Description
Operand access count (read/with cache)	OAR*	The number of times the operand access is performed on the cacheable area when the cache is enabled (read access only).
Operand access count (write/with cache)	OAW*	The number of times the operand access is performed on the cacheable area when the cache is enabled (write access only).
Operand access count (read and write/with cache)	OARW*	The number of times the operand access is performed on the cacheable area when the cache is enabled (both read and write accesses).
Internal RAM operand access count	OARAM	The number of times the internal RAM area is accessed.
All operand access count	OA	The number of all operand accesses.
Internal I/O area access count	IOA	The number of times the internal I/O area is accessed.
Operand cache read miss count	DCR	The number of times operand cache misses occur at data reading.
Operand cache write miss count	DCW	The number of times operand cache misses occur at data writing.
Operand cache read and write miss count	DCRW	The number of times operand cache misses occur at data reading or writing.
Instruction cache miss count	EC	The number of times instruction cache misses.
UTLB miss count	DT	The number of times UTLB misses occur at data access.
Instruction TLB miss count (ITLB and UTLB misses)	ET	The number of times UTLB and ITLB misses occur at instruction access.
Instruction cache miss count	EF*	The number of times instructions are fetched from the cacheable area when the cache is enabled.
All instruction fetch count	EA	The number of times all instructions are fetched.
Branch instruction execution count	B	The number of times branch instructions are issued (instructions to be counted: BF (other than displacement 0), BF/S and BT (other than displacement 0), BT/S, BRA, BRAF, and JMP).
Branch taken count	BT	The number of times branches are taken (branches to be counted are the same as mode B).
BSR/BSRF/JSR instruction execution count	BBJ	The number of times the BSR, BSRF, or JSR instruction is issued.
Instruction execution count	E	The number of times instructions are issued.

Table 2.11 Measurement Items (cont)

Event	Keyword	Description
Two-instruction concurrent execution count	E2	The number of times two instructions are issued at the same time.
FPU instruction execution count	EFP	The number of times FPU instruction is issued.
TRAPA instruction execution count	ETR	The number of times the TRAPA instruction is executed.
Interrupt count (normal)	INT	The number of interrupts (generally except for NMI).
Interrupt count (NMI)	NMI	The number of NMI interrupts.
UBC-A match count	UA	The number of times channel A of UBC is satisfied.
UBC-B match count	UB	The number of times channel B of UBC is satisfied.
Instruction cache-fill cycle	ECF	The number of instruction cache-fill cycles.
Operand cache-fill cycle	OCF	The number of operand cache-fill cycles.
Elapsed-time cycle	TM	The number of cycles for elapsed time.
Pipeline-freeze (by cache miss or instruction)	PFCF	Pipeline-freeze cycle due to instruction cache misses.
Pipeline-freeze (by cache miss or data)	PFCD	Pipeline-freeze cycle due to operand cache misses.
Pipeline-freeze (by branch instruction/interrupt)	PFB	Pipeline-freeze cycle due to branch instructions or exceptions.

Note: For the non-cache operand accesses due to the PREF instruction or TLB.c=0, the correct value cannot be counted.

The events can be counted even in the conditions shown in table 2.12, in addition to the normal count conditions.

Table 2.12 Performance Count Conditions

Event	Count Condition	Target Mode
Instruction cache miss count	• Includes instruction fetch for the cache-off area to count the number of times the instruction has not been fetched in one cycle. • When a cache miss occurs during an overrun fetch generated at exception.	EC
TLB miss count	When the TLB miss is canceled by an exception having a higher priority than that of the TLB miss	DT and ET
Instruction fetch count	When the instruction fetch request by the CPU is accepted.	EF and EA
Instruction issue count	Counts one when two instructions are issued at the same time.	E
	Counts one to three when instruction fetch exception (instruction address error, instruction TLB miss exception, or instruction TLB protection violation exception) occurs.	E and E2
FPU instruction issue count	• Counts one when two instructions are issued at the same time. • The following shows the FPU instructions: LDS Rm, FPUL, LDS.L @Rm+, FPUL, LDS Rm, FPSCR, LDS.L @Rm+, FPSCR, STS FPUL, Rn, STS.L FPUL, @-Rn, STS FPSCR, Rn, STS.L FPSCR, @-Rn Others: instructions that the instruction code is H'Fxxx	EFP
UBC satisfaction count	Also counts when the emulator uses the UBC as Break Condition 5,6.	UA and UB
Pipeline freeze due to cache miss	Includes the following freeze times: • At internal RAM or internal I/O space access • At instruction or operand access without cache	PFCE and PFCD
Pipeline freeze cycle due to branch instruction or exception	Counts only one cycle at branch instruction execution except when the delay slot instruction is executed with one-cycle delay. One instruction is executed in one cycle, which is similar to the branch count. When the instruction in the branch destination does not exist in the instruction cache, the delay after the second cycle is counted by the ECF. In the PFB, all branch instructions can be counted.	PFB

(c) Counting method

One of the following methods can be specified by each of measurement channels 1 and 2.

1. Counted by the CPU operating clock
2. Counted by the ratio of the CPU operating clock to the bus clock

When the above method 1 is specified, one CPU operating clock cycle is counted as one. When method 2 is specified, the count is incremented by 3, 4, 6, 8, 12, or 24, according to the clock frequency ratio (ratio of the CPU clock to the bus clock). In this case, the execution time can be calculated by the following expression:

$T = C \times B / 24$ (T: Execution time; B: Time of one bus clock cycle; C: Count)

When the ratio of the CPU clock to the bus clock is changed in the user program, it is recommended to select method 2, above, to count the number of cycles.

The following shows examples to measure the performance of the user program by the performance measurement function.

(i) Measuring cache hit ratio

Specify measurement channel 1 to count the cache misses (for data read and write) and specify measurement channel 2 to count operand accesses (read and write) to the cacheable area while the cache is enabled. Specify, with both the channels, the measurement from the start to the end of user program execution.

With the above command settings, the cache miss count and the access count to the cacheable area can be measured, and the cache hit ratio in the executed user program can be obtained.

(ii) Measuring ratio of execution time in specified program area to total execution time

Specify measurement channel 1 to measure the elapsed cycle count from the start to the end of user program execution. Specify measurement channel 2 to measure the elapsed cycle count during execution from the specified start PC to the specified end PC.

With both the channels, the total elapsed cycle and the elapsed cycle for the specified program area can be measured, and the ratio of the execution time in the specified program area to the total execution time can be obtained.

Notes: 1. The counter for performance measurement has 48 bits. A maximum of $2^{48} = 2.8 \times 10^{14}$ counts and 16-day cycles (when the CPU operating frequency is 200 MHz) can be measured. If a counter overflow occurs, the count becomes invalid.

2. For details on command-line syntax, refer to the online help.

2. Displaying the measured result

The measured result is displayed in the [Performance Analysis] window or the PERFORMANCE_ANALYSIS command with hexadecimal (32 bits).

Note: If a performance counter overflows as a result of measurement, “********” will be displayed.

3. Initializing the measured result

   To initialize the measured result, select [Initialize] from the popup menu in the [Performance Analysis] window or specify INIT with the PERFORMANCE_ANALYSIS command.

2.2.9 Interrupts

During emulation, any interrupt to the SH7751 can be used. Whether or not to process interrupts during emulator command execution or in command input wait state can be specified.

— When no interrupt is processed during user program break

  While the emulator is executing the user program or is in command input wait state, interrupts are not processed generally. However, if an internal interrupt or an edge sensitive external interrupt occurs in command input wait state, the emulator holds the interrupt and executes the interrupt processing routine when the GO command is entered.

— When interrupts are processed during user program break

  Use the INTERRUPT command to execute an interrupt during a user program break. This function is supported only with the command lines.

  - Execute only non-maskable interrupt (NMI)
  - Sets the priority and executes only interrupts with high priority

Notes: 1. Check that the interrupt handler operates normally before using this function. In addition, do not execute a non-limited loop or the sleep instruction in the interrupt handler. If the processing of the handler does not end, the emulator generates a Communication Timeout error.

2. When interrupts are accepted during user program break, user interrupt processing is not traced. In this case, continuous trace is not enabled.

3. Use the NOP instruction at the delay slot after the RTE instruction in the interrupt handler.

4. If a user interrupt is inserted while the user program breaks until the processing ends, do not set a BREAKPOINT in the interrupt handler. The emulator may generate a Communication Timeout error. Use the Break Condition function.

5. For details on command-line syntax, refer to the online help.

2.2.10 CPU Status Acquisition

The emulator can display the SH7751 status during user program execution in realtime. It displays the items selected in the [Extended Monitor Configuration] dialog box in the [Extended Monitor] window during user program execution. The emulator can display the state of the moment when a command is input for the specified register through the command-line function.

Notes: 1. This function is only valid during user program execution. If this function is used during a user program break, an undefined value is displayed.
2. A read value during reset is not guaranteed.
3. In the sleep or deep sleep mode, only the STATUS or FRQCR can be read.
4. The display can be updated in the interval between 1000 to 65535 ms.

Table 2.13 shows the details of the items that can be displayed.

Table 2.13 Display Status

Item	Example	Description
PC	H'A0000104	Displays the PC value.
SR	H'000000F0	Displays the SR register value.
FPSCR	H'000000F0	Displays the FPSCR register value.
INTEVT	H'00000100	Displays the INTEVT register value.
EXPEVT	H'00000600	Displays the EXPEVT register value.
FRQCR register	H'00000102	Displays the FRQCR register value.
MMUCR.AT	H'0	Displays the AT bit value in the MMUCR register.
ASID	H'01	Displays the ASID value in the PTEH register.
CCR	H'00000001	Displays the CCR register value.
SBUS	H'00000000	Displays the load/store bus address. (internal bus)
EBUS	H'0000000	Displays the external bus address.
SBTYPE	B'1101	Displays the internal bus state. Each bit has the following meanings: **Bit3: Bus access** 0: Without bus access 1: With bus access If bit 3 is 0, other bits of SBTYPE and all bits of SBUS are invalid.

Table 2.13 Display Status (cont)

Item	Example	Description

SBTYPE (cont)	B'1101	**Bit2: Read or write cycle**
		0: Read cycle
		1: Write cycle
		Bit1,0: Bus width
		Bit1=0, Bit0=0: 8-bit bus width
		Bit1=0, Bit0=1: 16-bit bus width
		Bit1=1, Bit0=0: 32-bit bus width
		Bit1=1, Bit0=1: 64-bit bus width

EBTYPE	B'0000000	Displays the external bus state. Each bit has the following meanings: **Bit5: Bus mode at DMA transfer** Displays an invalid value in the CPU access. 0: Burst mode 1: Cycle steal mode **Bit4: CPU access or DMAC access** 0: Access from CPU 1: Access from DMAC **Bit6,3,2: One transfer unit in DMA transfer** Bit6=0, Bit3=0, Bit2=0: 64 bits Bit6=1, Bit3=0, Bit2=0: 32 bytes Bit6=0/1, Bit3=0, Bit2=1: 8 bits Bit6=0/1, Bit3=1, Bit2=0: 16 bits Bit6=0/1, Bit3=1, Bit2=1: 32 bits These bits indicate memory access in the chip instead of the bus width. **Bit1: Read or write cycle** 0: Read cycle 1: Write cycle **Bit0: Bus access** 0: Without bus access 1: With bus access If bit 0 is 0, other bits of EBTYPE and all bits of EBUS are invalid. Note: When bit 0 is 1 and bit 4 is 0, bits 5 and 6 become invalid.
STATUS	B'00	Displays the STATUS pin state.

Table 2.13 Display Status (cont)

Item	Example	Description
Condition match flag	A=0	Displays whether the channel A condition of the UBC has been satisfied. When the UBC is used as a Break Condition, it displays whether Break Condition 6 has been satisfied. 0: Not satisfied 1: Satisfied
	B=0	Displays whether the channel B condition of the UBC has been satisfied. When the UBC is used as a Break Condition, it displays whether Break Condition 5 has been satisfied. 0: Not satisfied 1: Satisfied
	BC1=0	Displays whether Break Condition 1 has been satisfied. 0: Not satisfied 1: Satisfied
	BC2=0	Displays whether Break Condition 2 has been satisfied. 0: Not satisfied 1: Satisfied
	BC3=0	Displays whether Break Condition 3 has been satisfied. 0: Not satisfied 1: Satisfied
	BC4=0	Displays whether Break Condition 4 has been satisfied. 0: Not satisfied 1: Satisfied
Condition match flag for sequential break	A=0	When the sequential break condition of the UBC is selected, this bit is 1 when the channel A condition has been satisfied and the channel B condition has not been satisfied. When the UBC is used as a Break Condition, channel A and channel B correspond to Break Condition 6 and Break Condition 5, respectively. This bit is 1 when Break Condition 6 has been satisfied and Break Condition 5 has not been satisfied.
	BC4=0	When Sequential break condition 4-3-2-1 is selected, this bit is 1 when Break Condition 4 has been satisfied and Break Condition 3 has not been satisfied. It is also 1 when Break Condition 4 is satisfied again after Break Condition 3 has been satisfied.

Table 2.13 Display Status (cont)

Item	Example	Description
Condition match flag for sequential break (cont)	BC3=0	When Sequential break condition 4-3-2-1 and Sequential break condition 3-2-1 are selected, this bit is 1 when Break Condition 3 has been satisfied and Break Condition 2 has not been satisfied. It is also 1 when Break Condition 3 is satisfied again after Break Condition 2 has been satisfied.
	BC2=0	When Sequential break condition 4-3-2-1, Sequential break condition 3-2-1, and Sequential break condition 2-1 are selected, this bit is 1 when Break Condition 2 has been satisfied and Break Condition 1 has not been satisfied. It is also 1 when Break Condition 2 is satisfied again after Break Condition 1 has been satisfied.

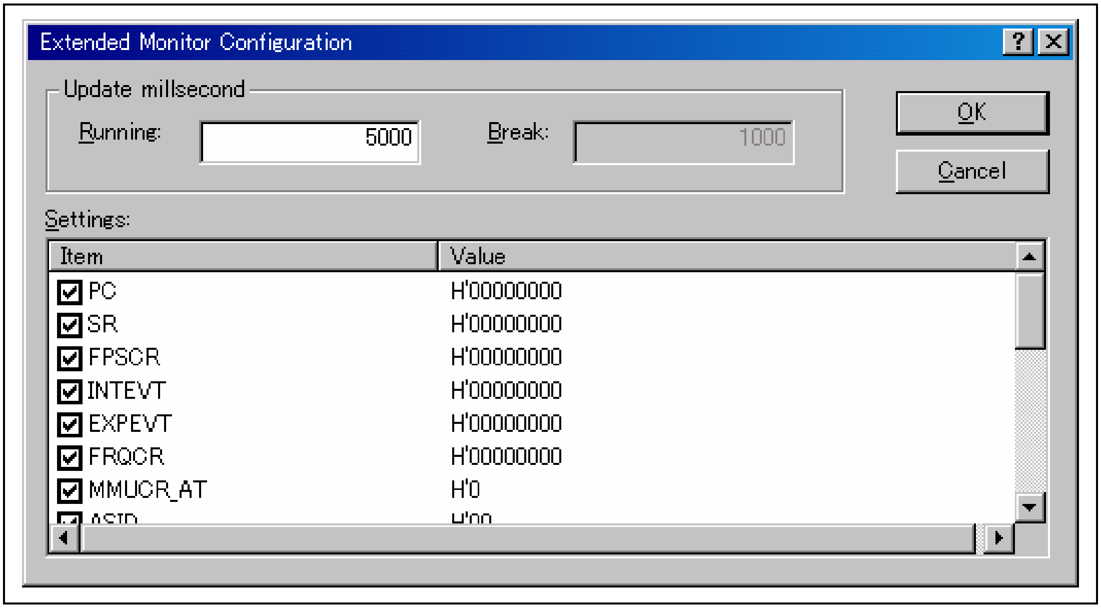
(a) Window function

To use the window function, the [Extended Monitor] window is displayed.

Open the [Extended Monitor] window by selecting [Display -> CPU -> Extended Monitor] or click the [Extended Monitor] toolbar button ( ).

Set the items to be displayed by selecting [Property] from the popup menu with the right mouse button to display the [Extended Monitor Configuration] dialog box.

Select the check boxes in the [Settings] group box for the items that are to be displayed.

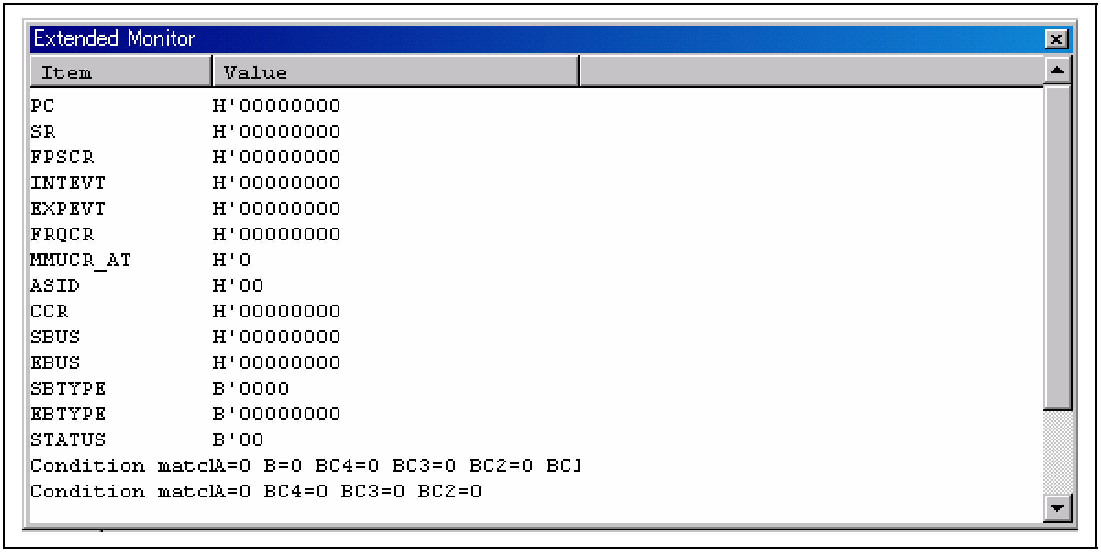


Running: 5000
Break: 1000

Item	Value
✓ PC	H'00000000
✓ SR	H'00000000
✓ FPSCR	H'00000000
✓ INTEVT	H'00000000
✓ EXPEVT	H'00000000
✓ FRQCR	H'00000000
✓ MMUCR_AT	H'0
✓ ASID	H'00

Figure 2.9 [Extended Monitor Configuration] Dialog Box

The items that have been selected are displayed in the [Extended Monitor] window.



Item	Value
PC	H'00000000
SR	H'00000000
FPSCR	H'00000000
INTEVT	H'00000000
EXPEVT	H'00000000
FRQCR	H'00000000
MMUCR_AT	H'0
ASID	H'00
CCR	H'00000000
SBUS	H'00000000
EBUS	H'00000000
SBTYPE	B'0000
EBTYPE	B'00000000
STATUS	B'00
Condition matc	A=0 B=0 BC4=0 BC3=0 BC2=0 BC1
Condition matc	A=0 BC4=0 BC3=0 BC2=0

Figure 2.10 [Extended Monitor] Window

Notes: 1. CPU status acquisition function [Condition match flag]:
The Break Condition function clears the condition match flag after a break occurred. Therefore, note that there are following limitations on measurement of this function.

Break Condition 1,4: Have meaning when they are used as the measurement start/end condition in the performance measurement function. In other cases, they are invalid in this product.

Break Condition 2,3,5,6: Are invalid in this product.

When Break Condition 5,6 are used as the UBC: The condition match flag is 1 when each channel in the UBC is satisfied until the flag is cleared.

2. CPU status acquisition function during standby:
The read value during standby cannot be guaranteed.

SuperH^{TM} Family E10A Emulator
Additional Document for User's Manual
Specific Guide for the SH7751 E10A Emulator

Publication Date:	Rev.1.00, December 16, 2003
Published by:	Sales Strategic Planning Div.
	Renesas Technology Corp.
Edited by:	Technical Documentation & Information Department
	Renesas Kodaira Semiconductor Co., Ltd.

SuperH™ Family E10A Emulator
Additional Document for User's Manual

Renesas Electronics Corporation

1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ10B0080-0100H