Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010 Renesas Electronics Corporation

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

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

Notice

- 1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
- Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
- 3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
- 4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
- 5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
- 6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
- 7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
 - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
 - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anticrime systems; safety equipment; and medical equipment not specifically designed for life support.
 - "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
- 8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
- 9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
- 10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
- 11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
- 12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.
- (Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majorityowned subsidiaries.
- (Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

High-performance Embedded Workshop

Real-time OS aware debugging

Summary

This document describes how to use the real-time OS aware debugging making use of High-performance Embedded Workshop.

The real-time OS aware debugging can be used for the following real-time OSes.

Corresponding Real-Time OS:

- HI7200/MP
- HI7750/4
- HI7700/4
- HI7000/4
- RI600/4
- M3T-MR30/4
- HI1000/4
- Smalight OS
- ThreadX
- NORTi
- osCAN
- μC/OS-II
- embOS
- μT-Kernel
- TOPPERS/ASP
- TOPPERS/JSP

Contents

1.	Introd	luction	3
2.	Overv	view of Real-time OS aware debugging	4
	2.1	Operating Environment	4
3.	Quick	Start Guide	7
	3.1 3.2 3.3 3.4 3.5 3.6	Downloading a Program using Real-time OS Selecting the OS Definition File Executing the Program Referring to the Task State Displaying the Task Execution History Displaying the Task Execution Time.	8 9 10 11
4.	Funct	ions	. 12
	4.1 4.2	List of Functions Functional Descriptions	
5.	Notes		.28
6.	FAQ.		. 30
7.	Webs	ite and Support	.31

1. Introduction

This application note is for customers using the real-time OS aware debugging of High-performance Embedded Workshop.

This document describes the functions using the following environment. It is assumed that the real-time OS and High-performance Embedded Workshop have been installed in the host PC in advance. For the installation method, refer to the manual for each product.

[Environment in which operation was confirmed]

- (1) Host PC: Windows Vista®, Windows®XP, Windows®2000
- (2) High-performance Embedded Workshop V.4.06.00
- (3) Real-time OS for SuperH Family manufactured by Renesas Technology Corp. [HI7000/4 V.2.02 Release 05]
- (4) SuperH Family RISC engine simulator/debugger V.9.09.00

2. Overview of the Real-time OS aware debugging

The real-time OS aware debugging are functions for supporting task-level debugging of the real-time OS application on High-performance Embedded Workshop. High-performance Embedded Workshop incorporates these functions as standard functions. The real-time OS does not have to be built again and it does not have an overhead. (Debug patches are unnecessary.)

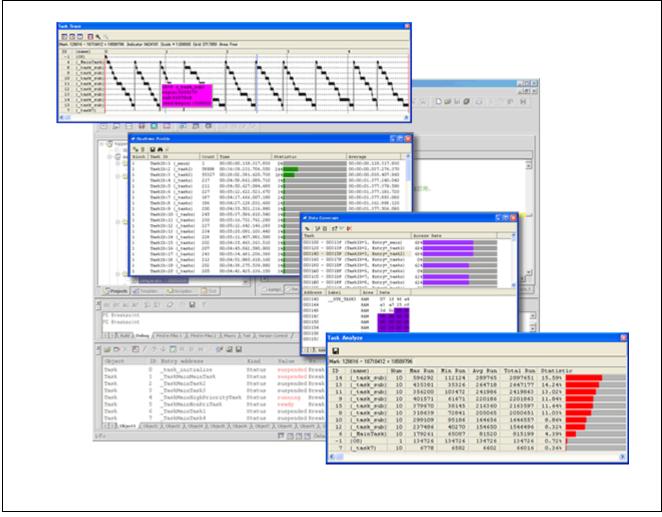


Figure 2.1 Real-time OS aware debugging

2.1 Operating Environment

The operating environment of the real-time OS aware debugging is as follows:

- Host machine
 Windows Vista®, Windows®XP, Windows®2000
- High-performance Embedded Workshop
 V.4.06.00 or later

• Corresponding Real-Time OS

Table 2.1 List of Corresponding Real-Time OS

Real-time OS for SuperH Family manufactured by Renesas [HI7200/MP]V.1.00 Release 03 or laterSH2A-DUALReal-time OS for SuperH Family manufactured by Renesas [HI7750/4]V.2.02 Release 04 or laterSH-4, SH-4AReal-time OS for SuperH Family manufactured by Renesas [HI7700/4]V.2.03 Release 02 or laterSH4AL-DSP, SH-3, SH3-DSReal-time OS for SuperH Family manufactured by Renesas [HI7700/4]V.2.02 Release 04 or laterSH4AL-DSP, SH-3, SH3-DSReal-time OS for SuperH Family manufactured by Renesas [HI7000/4]V.2.02 Release 04 or laterSH-2, SH2-DSP, SH-2A, SHReal-time OS for RX Family manufactured by Renesas [RI600/4]V.1.00 Release 00 or laterRX Family*Real-time OS for M16C Series, R8C Family manufactured by Renesas [M3T-MR30/4]V.4.00 Release 00 or laterM16C/60, M16C/50, M16C/3 M16C/20, M16C/10, M16C/10	
manufactured by Renesas [HI7750/4]V.2.03 Release 02 or laterSH4AL-DSP, SH-3, SH3-DSReal-time OS for SuperH Family manufactured by Renesas [HI7700/4]V.2.02 Release 04 or laterSH-2, SH2-DSP, SH-2A, SHReal-time OS for SuperH Family manufactured by Renesas [HI7000/4]V.2.02 Release 04 or laterSH-2, SH2-DSP, SH-2A, SHReal-time OS for RX Family manufactured by Renesas [RI600/4]V.1.00 Release 00 or laterRX Family*Real-time OS for M16C Series, R8C Family manufactured by Renesas [M3T-MR30/4]V.4.00 Release 00 or laterM16C/60, M16C/50, M16C/30, M16C/20, M16C/10, M16C/70, M16C/10, M16C/70, M16C/20, M16C/10, M16C/70,	
manufactured by Renesas [HI7700/4] V.2.02 Release 04 or later SH-2, SH2-DSP, SH-2A, SH Real-time OS for SuperH Family manufactured by Renesas [HI7000/4] V.2.02 Release 04 or later SH-2, SH2-DSP, SH-2A, SH Real-time OS for RX Family manufactured by Renesas [RI600/4] V.1.00 Release 00 or later RX Family* Real-time OS for M16C Series, R8C Family manufactured by Renesas [M3T-MR30/4] V.4.00 Release 00 or later M16C/60, M16C/50, M16C/3	
manufactured by Renesas [HI7000/4] Real-time OS for RX Family manufactured by Renesas [RI600/4] V.1.00 Release 00 or later RX Family* Real-time OS for M16C Series, R8C Family manufactured by Renesas [M3T-MR30/4] V.4.00 Release 00 or later M16C/60, M16C/50, M16C/3 M16C/20, M16C/10, M16C/70 M100 M100 M100 M	Р
by Renesas [RI600/4] Real-time OS for M16C Series, R8C Family manufactured by Renesas [M3T-MR30/4] V.4.00 Release 00 or later M16C/60, M16C/50, M16C/3 M16C/20, M16C/10, M16C/7 R8C Family	-1
Family manufactured by Renesas M16C/20, M16C/10, M16C/T [M3T-MR30/4] R8C Family	
	0, ïny Series
Real-time OS for H8SX, H8S Family V.1.05 Release 01 or later H8SX Family	
manufactured by Renesas [HI1000/4] H8S Family	
Smalight OS manufactured by Renesas V3.10 or later SH-2	
Northern Japan Semiconductor, Inc. RX Family*	
M16C/60, M16C/30, M16C/T	iny Series
R8C Family	
H8SX Family	
H8S Family	
H8 Family	
ThreadX manufactured by Express Logic, G5.1.5.0 SH-2A	
NORTi manufactured by MiSPO Co., Ltd. Version4 Release 1.20 SH-4A, SH-2A	
Version4 Release 1.14 H8SX Family	
H8S Family	
osCAN manufactured by Vector Informatik Version 3.05 M16C/60, M16C/Tiny Series	
GmbH R8C Family	
μC/OS-II manufactured by Micrium V2.86 (AN-Renesas-SH7201) SH-2A	
V2.89 RX Family*	
V2.83 (AN-1662) M16C/60 Series	
V2.86 H8SX Family	
(AN-Renesas-H8S2215R, AN-Renesas-H8S2472, AN-Renesas-H8SX1664)	
embOS* manufactured by SEGGER Version 3.80g SH-2A	
μT-Kernel by T-Engine Forum v1.00.00 M16C/62P Group	
TOPPERS/ASP by TOPPERS Project Release 1.3.1 SH-2A	
Release 1.3.2 M16C/60 Series	
TOPPERS/JSP by TOPPERS Project Release 1.4.3 H8/3048 Group	

Debugger	Supported Microcomputers	Debugger Version	Support Status
Simulator/debugger	SuperH Family	V.9.08.00 or later	Already
	RX Family*	V.1.00.00 or later	supported
	H8SX, H8S, H8 Family	V.5.07.00 or later	
E1 emulator debugger	RX Family*	—	Planned to be supported
E20 emulator debugger	RX Family*	—	Planned to be supported
E10A-USB emulator debugger	SuperH Family, H8SX Family, H8S Family	V.3.00 Release 00 or later	Already supported
	SH2A-DUAL, SH4A-MULTI	—	Planned to be supported
E8a emulator debugger	M16C Series, R8C Family, H8 Family	V.1.02 Release 00 or later	Already supported
E100 emulator debugger	M16C Series, R8C Family, H8SX Family	V.1.00 Release 00 or later	Already supported
E200F emulator debugger	SH-2A	—	Planned to be supported

Corresponding Debuggers Table 2.2 List of Corresponding Debuggers

• Corresponding Compilers

- -C/C++ compiler package for H8SX, H8S, H8 Family V.6.02 Release 00 or later

Note: * Supported by High-performance Embedded Workshop V.4.07.00 or later.

3. Quick Start Guide

This section describes how to use the basic functions (refer to the object state, display the task execution history, etc.) of the real-time OS aware debugging of High-performance Embedded Workshop.

Workspaces of High-performance Embedded Workshop using the real-time OS should be prepared in advance. For the method of creating a workspace, refer to the manual or online help of both the real-time OS and High-performance Embedded Workshop.

In the subsequent descriptions, the real-time OS "HI7000/4" and the SuperH Family simulator/debugger which are both manufactured by Renesas Technology Corp. are used. The real-time OS aware debugging is used in a similar manner even when using other real-time OS products or debuggers.

3.1 Downloading a Program using Real-time OS

Launch High-performance Embedded Workshop, open a workspace (a file whose extension is "hws") using HI7000/4, and select [Download Modules] from the [Debug] menu in High-performance Embedded Workshop to download programs to the simulator/debugger.

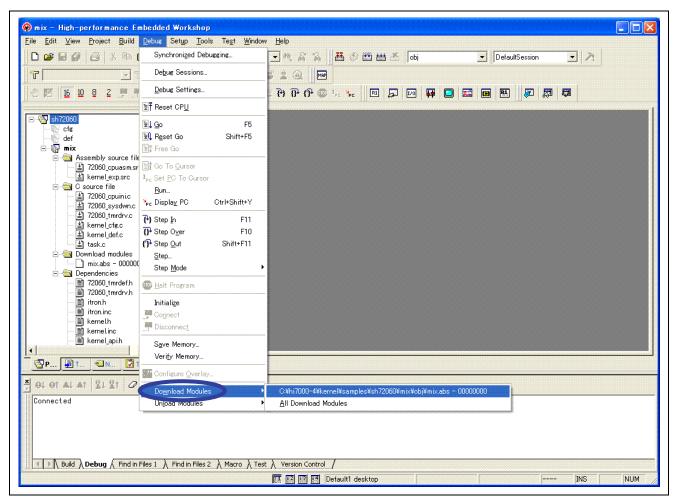


Figure 3.1 Downloading a Program

3.2 Selecting the OS Definition File

When program download is executed, the Select OS Definition File dialog box is automatically displayed. Select the relevant real-time OS and click the [OK] button.

Select OS Definition File		
HI7000/4 V2, HI7700/4 V2, HI7750/4		
	Cancel	
	Not Select	
< ni		
T Never show this dialog box at dowr	load	

Figure 3.2 Selecting the OS Definition File

3.3 Executing the Program

Select [Reset Go] from the [Debug] menu to execute the program. Then, to complete initialization of the HI7000/4 kernel, wait for a specific period before selecting [Halt Program] from the [Debug] menu to stop the program.

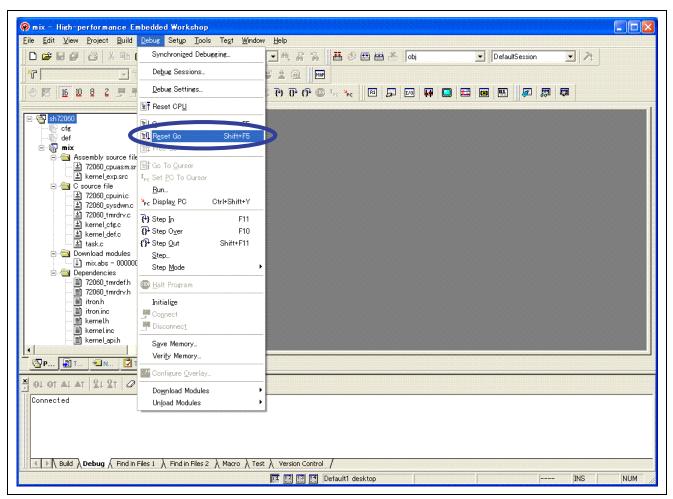


Figure 3.3 Executing the Program

High-performance Embedded Workshop Real-time OS aware debugging

🏟 mix – High-performance Ei			
	<u>Debug</u> Set <u>up</u> <u>T</u> ools Te <u>s</u> t <u>W</u> indow Synchronized Debugging		
		▼ 翰 解 解 III 番 多 陞 描 述 Iobi I DefaultSession I 入	
	Debug Sessions		
0 10 16 10 8 2 💻	Debug Settings	토 (권) (권) (전) 🥯 (Jac Sac) 🔟 💭 📖 🐺 🔲 🔛 📖 🛤 🛛 💭 💭	
	E¶ Reset CP <u>U</u>		
	<u>≣</u> ↓ <u>G</u> o F5		
def]] Reset Go Shift+F5]] Free Go		
🖻 🖓 mix E 🏐 Assembly source file			
72060_cpuasm.sr kernel_exp.src	'≣' Go To <u>C</u> ursor I _{PC} Set <u>P</u> C To Cursor		
🖻 🔄 C source file	<u>R</u> un		
72060_cpuini.c 72060_sysdwn.c	א _{PC} Displa <u>y</u> PC		
- 말 72060_tmrdrv.c - 말 kernel_cfg.c	🖓 Step In 🛛 F11		
– – ≚] kernel_def.c	Or Step Oyer F10		
task.c ⊡ — ∰ Download modules	{∯ Step <u>O</u> ut Shift+F11 <u>S</u> tep		
🗔 💭 mix.abs - 000000	Step <u>M</u> ode	•	
⊡ — Dependencies — 72060_tmrdef.h			
≣ 72060_tmrdrv.h	Initialize		
- 📄 itron.inc	Connect		
kernel.h	Disconnec <u>t</u>		
- 📄 kernel_api.h	S <u>a</u> ve Memory		
4 ──────────────────────────────────	Veri <u>f</u> y Memory		
	Configure <u>O</u> verlay		
× 01 OT AL AT 21 21 0	Do <u>w</u> nload Modules	•	
Connected	Un <u>l</u> oad Modules	•	
		<u> </u>	
Build Debug / Find in	Files 1 λ Find in Files 2 λ Macro λ Test		
		III III Default1 desktop INS NUM	11.

Figure 3.4 Halting the Program

3.4 Referring to the Task State

Select [RTOS]->[OS Object] from the [View] menu to open the OS Object window in which the task state can be referenced. In this example, it is obvious that task ID 2 is in the RUNNING state.

$\square \times \mathbb{N}$	/ 个 🗸 🖸	R B N 🖪 🛃 📮	🖬 🖻		
Object	ID	Entry address	Kind	Value	Auto update
Task	1	MainTask	Status(WaitFactor)	DORMANT	Break
Task	2		Status (WaitFactor	RUNNING	Break
Task	3		Status(WaitFactor)	NON-EXISTENT	Break

Figure 3.5 Referring to the Task State

3.5 Displaying the Task Execution History

The Task Trace window displays the task execution history using the trace data of the simulator/debugger or emulator debugger. To implement the trace settings, open the Task Trace window once by selecting [RTOS]->[Task Trace] from the [View] menu. After that, execute the program again. Thus, the task execution history will be displayed in the Task Trace window. In this example, it is obvious that task ID 1 and task ID 2 are operating.

	🔲 🖬 🔍						
Mark: 169	- 10385 = 1021	6 Indicator: 6	i081 Scale: * 1.0	000000 Grid: 2043 /	Area: Full		
		0	1	2	3	4	
	(OS[Idle]) (MainTask						
and and and along all along the	(task7)						

Figure 3.6 Displaying the Task Execution History

3.6 Displaying the Task Execution Time

The Task Analyze window opened by selecting [RTOS]->[Task Analyze] from the [View] menu displays the task execution time and CPU occupancy taken up by it using the trace data of the simulator/debugger or emulator debugger.

ark: 1	69 - 10390 = 10221							
ID	(name)	Num	Max Run Time	Min Run Time	Avg Run Time	Total Run Time	Scatistic	
-1	(OS[Idle])	1	5917	5917	5917	5917	57.89%	
1	(MainTask)	2	2465	1201	1833	3666	35.87%	
2	(task7)	1	638	638	638	638	6.248	

Figure 3.7 Displaying the Task Execution Time

4. Functions

This section describes each function in detail.

4.1 List of Functions

Table 4.1 lists the functions that can be used by each real-time OS. Table 4.2 lists the functions supported in each debugger.

Function	HI7200/MP, HI7750/4, HI7700/4, HI7000/4	RI600/4, M3T-MR30/4	HI1000/4	Smalight OS	ThreadX, embOS, TOPPERS/JS P	NORTi, μC/OS-II, μT-Kernel	osCAN, TOPPERS/ASP
OS object	V	V	V	V	V	V	v
Task trace	V	V	V	V	V	v	V
Task analyze	V	V	V	V	V	V	 ✓
Task step	—	V	v	V	—	_	_
Currently executed task display	_	~	V	V	—	V	V
Execution history text display	_	V	V	V	_	V	 ✓
Real-time profile		 ✓ 	V	V	_	 ✓ 	 ✓
Task specification for each condition	_	~	v	V	_	V	V
Stack coverage		V		V	_		
Memory protection	_	~		V		_	_
Dispatch detection	_	V	V	V	—	_	—

 Table 4.1
 Function Relationship (Real-time OS)

Function	Simulator	E1	E20	E10A-USB	E8a	E100	E200F
OS object	~	V	V	~	V	(RAM monitoring supported)	~
Task trace	V	~	(Large capacity)	×	_	(Large capacity)	V
Task analyze	V		(Large capacity)	*	—	(Large capacity)	V
Task step	_	~	V	_	_	V	—
Currently executed task display	—	~	V	—		 ✓ 	—
Execution history text display	—		—	—	_	V	—
Real-time profile	-	—	—	—	_	V	—
Task specification for each condition	—		_	—	—	V	—
Stack coverage	—	_	_	—	—	V	—
Memory protection	—		—	—	—	V	—
Dispatch detection	—		—	—	—	V	—

Table 4.2 Function Relationship (Debugger)

Note: * Use a 36-pin or 38-pin cable (14-pin cable cannot be used). The time is not displayed (except for SH2A-DUAL).

- 4.2 Functional Descriptions
- 4.2.1 OS Object Window (Reference of Object State)

				1	
🥔 OS Obj	set				
😹 🗗 🗙		/ ↑ ↓ 🖾 R B	N 🗈 📝 🔛 🖬		
Object	ID	Entry address	Kind	Value	Auto update RAM monitor
maak	1		Ob a burg	DODUU IN	Durch
Task	2	_task1	Status	WAIT	Break
Task	3	Cask4	Status	RUN	break
Task	4	_taskx	Status	DORMANT	Break
Task	5	taskx	Status	DORMANT	Break
Task	1	main	WaitFactor		Break
Task	2	task1	WaitFactor	Semaphore	Break
Task	3	task2	WaitFactor		Break
Task	4	taskx	WaitFactor		Break
Task	5	tasky	WaitFactor		Break
Semaphor	e 1		SemaphoreCount	0	Break
	ct1 /	Object2 λ Object3 λ C	bject4 λ Object5 λ Ot	piect λ Object7	7) Object8) Object9) Object10 /
	A		., N, N	, /,	

Figure 4.1 OS Object Window

How to Open

- Select [View->RTOS->OS Object].
- Click the [OS Object] toolbar button.

Description

The OS Object window displays the state of OS objects, such as tasks and semaphores.

When this window is opened for the first time, the states of all tasks are displayed automatically. Add display items using [Add objects] in the popup menu as required. The number of display items and the displayed contents differ in each OS.

When the object has an entry address (task, handler, etc.), the [Editor] window can be opened from the popup menu. If the displayed contents are updated, the updated values are displayed in red.

If RAM monitoring is specified for the area of the object values, [In] is displayed in the [RAM monitor] column.



- Real-time performance is not achieved while this window is opened because memory accesses will occur.
- The window may not be displayed correctly when execution is halted in the midst of program execution or OS kernel execution.

Popup Menu

The usable options are shown below.

Add Object ((Δ)	Displays the [Select Object Data] dialog box.			
	, · · · ·	The added OS object is inserted before the selected line.			
Delete Object	ct (D)	Deletes the selected OS object.			
Delete All Ol	bjects (L)	Deletes all OS objects in the currently displayed sheet.			
Edit Object (E)	Displays the [Select Object Data] dialog box. Another OS object can be selected here.			
Up (U)		Moves the selected OS object one line up.			
Down (D)		Moves the selected OS object one line down.			
Refresh (R)		Forces a manual update of the displayed contents.			
Auto Update (U)	Update During Execution (A)	Updates the displayed contents while the user program is being executed and when it has stopped.			
(-)	Update Only At Stop (B)	Updates the displayed contents when the user program has stopped.			
	No Update (L)	Does not update the displayed contents.			
Display Sou	rce (W)	Opens the [Editor] window corresponding to the entry address.			
Select All (A)	Selects all lines.			
Save (S)	Save Displayed Contents (D)	Saves the displayed contents of all sheets in a file.			
	Save Display Items (I)	Saves the display items of all sheets in a file.			
Read (L)		Reads the settings of all sheets from a file.			
Display Tool	bar	Switches between display and nondisplay of the toolbar.			
Customize T	oolbar	Customizes the toolbar buttons.			

4.2.2 Task Trace Window (Graphical Display of Task Execution History)

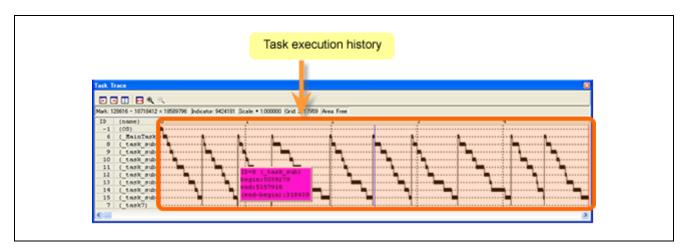


Figure 4.2 Task Trace Window

How to Open

- Select [View->RTOS->Task Trace].
- Click the [Task Trace] toolbar button.

Description

The Task Trace window graphically displays the result of measuring the task execution history of a program using the real-time OS. The items to be displayed are as follows:

Item	Description
ID	Displays the task ID.
(name)	Displays the task entry label name.

Moving the mouse to each information shown in the window opens a popup menu for displaying detailed information.

The status bar displays the following information.

- Time value of starting point marker's location
- Time value of ending point marker's location
- Duration between starting point and ending point markers
- Time value of current position marker's location
- Display magnification
- Duration of grid line interval
- Measurement (trace) range

Grid lines are displayed with the starting point marker as the base point.

Scales are displayed with the time where the starting point marker is located as 0, the left side (forward in terms of time) as negative, and the right side (backward in terms of time) as positive.

The generation cycle and processing time of interrupts can be roughly grasped by grid lines.

The duration of the displayed grid line interval is shown in the "Grid" area of the status bar.

The time values in the Task Trace window all mean the execution elapsed time with the program execution start point assumed to be 0.

In contrast, the numbers at the upper side of the grid lines (scales) in the Task Trace window are relative values (grid line interval is specified in the Value dialog box) with the starting point marker assumed to be 0, and they have nothing to do with time values (they are to facilitate visualization of the window).

Popup Menu

The usable options are shown below.

Starting Point Marker		Moves the starting point marker to the display area.			
Ending Point Marker		Moves the ending point marker to the display area.			
Current Position Mark	ker	Moves the current position marker to the display area.			
Adjust Display Magni	fication	Displays the range between the starting point and ending point markers using the entire horizontal width of the window.			
Enlarge Display Magnification		Enlarges the display magnification.			
Reduce Display Magnification		Reduces the display magnification.			
Halt Measure		Halts trace measurement and displays the result.			
Re-measure		Re-measures the trace data.			
Measurement Range Condition	Full	Stops trace acquisition when the trace memory has become full.			
	Free	Continues trace acquisition until program execution is stopped.			
Set		Sets the grid line interval and display magnification.			
Display Toolbar		Switches between display and nondisplay of the toolbar.			
Customize Toolbar		Customizes the toolbar buttons.			

4.2.3 Task Analyze Window (Graphical Display of Task Execution Time)

Task A	inalyze							
				\ \				
_	28616 - 18718412	= 18589	796		-			
ID	(name)	Num	Max Run	Min Run	Avg Run	Total Run	Statisti	с
14	(_task_sub	10	596292	112124	289765	2897651	15.59%	
13	(_task_sub	10	435381	35326	264718	2647177	14.24%	
11	(_task_sub	10	356200	103472	241986	2419863	13.02%	
9	(_task_sub		401571	61671	220186		11.84%	
15	(_task_sub		379670	38145	216360	2163597	11.64%	
8			318639	72841	205065		11.03%	
10	(_task_sub		299109	95186	164656	1646557	8.86%	
12	(_task_sub		237486	40270	154650	1546496	8.32%	
6	(_HainTask		179261	65087	81520	815199	4.39%	
-1	(OS)	1	134726	134726	134726	134726	0.72%	
7	(task7)	10	6778	6582	6602	66016	0.365	

Figure 4.3 Task Analyze Window

How to Open

- Select [View->RTOS->Task Analyze].
- Click the [Task Analyze] toolbar button.

Description

The Task Analyze window displays the status of CPU occupation.

The Task Analyze window functions in collaboration with the Task Trace window.

The Task Analyze window displays the result of statistically processing the data measured for the range specified by the starting point and ending point markers of the Task Trace window.

Clicking the maximum execution time and minimum execution time display areas in each line enables searching for the processing history of the maximum execution time and minimum execution time of the task corresponding to the clicked line.

The search result is indicated by the current position marker of the Task Trace window moving to the target location.

Popup Menu

The usable options are shown below.

Save In File (S)	Saves the displayed contents in a file in the text format or CSV format.
Initialize Display Order (D)	Returns the display order of the tasks to the way they were displayed first. The display order in the beginning is the order in which the tasks were executed.
Display Toolbar	Switches between display and nondisplay of the toolbar.
Customize Toolbar	Customizes the toolbar buttons.

4.2.4 Task Step (Step Execution Focused on a Specific Task)

• [Debug]->[RTOS Debug]->[Go To Cursor]

Executes the current task until it reaches the cursor's position.

The user program is executed from the current PC value and execution is continued until the PC value reaches the address indicated by the current text cursor's position (not the mouse cursor). If the task at execution stop differs from the task at execution start, execution will continue until the tasks become the same.

• [Debug]->[RTOS Debug]->[Step In]

Steps into the current task.

A single block of the user program is executed and then execution stops. The size of this block is normally a single instruction but the user can specify multiple instructions or a C/C++ source line (Step...).

When a subroutine is called, execution is halted after entering the subroutine, and the subroutine code is displayed.

If the task at execution stop differs from the task at execution start, execution will continue until the tasks become the same.

When step-in execution is performed to a system call, processing may not end correctly. In such a case, perform step-over execution.

• [Debug]->[RTOS Debug]->[Step Over]

Steps over the current task.

A single block of the user program is executed and then execution stops. The size of this block is normally a single instruction but the user can specify multiple instructions or a C/C++ source line (Step...).

When a subroutine is called, the subroutine is not entered, and the user program continues to be executed until the current PC location moves to the next line of the current display. If the task at execution stop differs from the task at execution start, execution will continue until the tasks become the same.

- [Debug]->[RTOS Debug]->[Step Out]
 - Steps out of the current task.

The user program is executed until the end of the current function is reached, and execution is stopped with the PC set to the next line of the calling function.

If the task at execution stop differs from the task at execution start, execution will continue until the tasks become the same.

4.2.5 Currently Executed Task Display in Status Bar

The task ID of the currently executed task is automatically displayed in the status bar.

4.2.6 Text Display of Task Execution History

	₩ ▲ :	ARGON 200000 12 19-	I DESCRIPTION OF THE OWNER OWNER OF THE OWNER OWNER OF THE OWNER													
• TaskID		*****	***********	***********												
TaskID																
1	= 2 (t															· · ·
1		(OF0368)	OX	1009	pi de la c											
		and the state of the														
1 · · · · · · · ·		(010368)														
· · · · · · · · · · · · · · · · · · ·		25C) <- 0	POPC4	6.65												
TaskID																
1	. ≓. ≊ . (_t	ask2)														
🗄 TaskID																
TaskID	i=;z.(_t	ask1)	alay opp	20144	Addes		ore la									
TaskID	= Z (_t 985, 000000	aski) 100 File: Cy														
TaskID Range:-00036 Cycle	= 2 (_t 985, 000000 Label	ask1) 100 FMe: Cy Address	Data	BUS	BHE	BIU	R/W I	RWT C	PU QI			TID				18)
TaskID Range: -00036 Cycle -0002646	985, 000000 Label	ask1) 100 FMe: Cy Address OFO36A	Data 0400	BUS 16b	BHE O	BIU IW	R/W I R	RNT C	1911 QN 101 Q	BUSA 1	Debug 1	2 (_task1)	00:00:00	670.329.	550	18)
Rangs:-00036 Cycle -0002646 -0002646	= 2 (_t 985, 000000 Label 5	ask1) 00 File: Cy Address 0F036A 0F036C	Data 0400 4075	BUS 16b 16b	BHE O O	BIU IW IW	R/W I R R	RWT C D C D P	PU Q1 10 2 10 2	BUSA 1	Debug 1 1	2 (task1) 2 (task1)	00:00:00	670,329, 670,393,	550 2,50	18) [•
Range: -00036 Cycle -0002646 -0002646 -0002646	985, 000000 Label 5 4	ask1) 00 File: Cy Address 0F036A 0F036C 0F036E	Data 0400 4075 8078	BUS 16b 16b 16b	BHE O O O	BIU IW IW	R/W I R R R	RWT C D C D P D C	200 Q1 00 2 00 2 00 2	BUSA 1 1	Debug 1 1 1	2 (<u>task1</u>) 2 (<u>task1</u>) 2 (<u>task1</u>)	00:00:00 00:00:00 00:00:00	670.329. 670.393. 670.456.	550 250 940	13)
Range: -00036 Cycle -0002646 -0002646 -0002646 -0002646	= 2 (_t 985, 000000 Label 5 5 4 3\$B_	ask1) 00 File: Cy Address 0F036A 0F036C 0F036E 000400	Data 0400 4075 8078 0032	BUS 16b 16b 16b 16b	BHE O O O O	BIU 100 100 100 100	R/W I R B R B W B	RWT C 0 C 0 P 0 C 0 -	PU Q3 100 2 100 2 100 2 100 2 100 2	BUSA 1 1 1 1	Debug 1 1 1 1	2 (task1) 2 (task1) 2 (task1) 2 (task1) 2 (task1)	00:00:00 00:00:00 00:00:00 00:00:00	670.329. 670.393. 670.456. 670.520.	550 250 940 690	15)
TaskID Range: -00036 Cycle -0002545 -0002646 -0002646 -0002646 -0002646 -0002646	= 2 (_t 985, 000000 Label 5 4 3\$B_ 2	ask1) 00 Fie: Cy Address 0F036A 0F036C 0F036E 000400 0F0370	Data 0400 4075 8078 0032 0F73	BUS 16b 16b 16b 16b 16b	BHE O O O O O O	BIU 100 100 100 100 100 100	R/W I R R W R	RWT C 0 R 0 R 0 C 0 -	190 Q3 100 2 100 2 100 2 100 2 100 2	BUSA 1 1 1 1 1	Debug 1 1 1 1 1 1	2 (task1) 2 (task1) 2 (task1) 2 (task1) 2 (task1) 2 (task1)	00:00:00 00:00:00 00:00:00 00:00:00 00:00:	670.329 670.393 670.456 670.520 670.584	550 250 940 680 400	13)
Rangs: -00036 Cycle -0002646 -0002646 -0002646 -0002646	e = 2 (_t 985, 000000 Label 5 4 3\$B_ 2 1	ask1) 00 File: Cy Address 0F036A 0F036C 0F036E 000400	Data 0400 4075 8078 0032	BUS 16b 16b 16b 16b 16b 16b	BHE O O O O O O	BIU 100 100 100 100	R/W I R (R (W (R (N (RWT C 0 P 0 P 0 C 0 - 0 C	PU Q5 50 2 50 2 50 2 50 2 50 2 50 2 50 0	BUSA 1 1 1 1 1 1 1	Debug 1 1 1 1 1 1	2 (task1) 2 (task1) 2 (task1) 2 (task1) 2 (task1) 2 (task1) 2 (task1)	00:00:00 00:00:00 00:00:00 00:00:00 00:00:	670.329. 670.393. 670.456. 670.520. 670.584. 670.648.	550 250 940 680 400 150	13)

Figure 4.4 Trace Window

How to Open

• Select [Display Execution History->Display Function Execution History] from the popup menu of the [Trace] window.

(The upper-stage window is displayed.)

• Select [Analyze Execution History] from the popup menu of the [Trace] window. (The analysis result is displayed in the upper-stage window.)

Description

The task execution history can be displayed only when debugging real-time OS programs.

To display the task execution history, the task ID needs to be selected in the [Options] page of the [Set Trace Conditions] dialog box which can be displayed from the [Set Trace...] menu.

The task execution history of the trace result is shown in the upper-stage window. The default is nondisplay.

When [Analyze Execution History] is selected from the popup menu of the upper-stage window, analysis of function execution history is started and the result is displayed.

If the branch destination address is not a function or if there is no corresponding debugging information for that address, an address is displayed (label and address are displayed when a label is set) instead of a function name.

The lower-stage window displays the trace result from the cycle in which the task selected in the upper-stage window was called.

<Display format of task execution history>

(1) The following is displayed when the task ID changes.

TaskID = Task ID number (entry name corresponding to task ID)

Example: TaskID = 1 (_main)

A function called within a task is displayed in the following format.
 Function name (function start address) <- Address of function caller

🛕 Note

The task execution history cannot be displayed when extracting or deleting conditions for trace are specified.

Upper-stage window:

Displays the execution history for individual tasks.

For the task execution history, the functions called within a task are not displayed in the tree structure. These functions only appear in the display of the execution order of the functions.

Expanded display or collapsed display can be selected for each task. The default is expanded display.

When analysis takes time, it can be canceled from the progress dialog box. When canceled, the task execution history is displayed up to the point where analysis was completed.

When the task entry label is double-clicked in the upper-stage window, the bus information on the point where that task was called is displayed in the lower-stage window.

Lower-stage window:

Displays the trace result from the cycle of the task selected in the upper-stage window.

Disassembly display, source display, and mixed display are also possible.

Popup Menu

The options usable in the upper-stage window are shown below.

Analyze Execution History	Analyzes the execution history of functions and displays the analysis result in the upper-stage window. Functions called within a task are not displayed in the tree structure. They only appear in the displayed function execution order.
Execution History Nondisplay	Closes the upper-stage window. The analysis result is cleared.
Function Search	Displays the [Function Search] dialog box. The function name specified in this dialog box will be searched for.
Function Forward Search	Searches forward from the line in which the function name specified in the [Function Search] menu was selected.
Function Backward Search	Searches backward from the line in which the function name specified in the [Function Search] menu was selected.
Jump To List	Displays the trace result from the cycle of the selected function.

4.2.7 Real-time Profile Window (Display of CPU Occupancy for a Non-Stop Long Period)

🛷 Realt	ime Profile				
°a 8	B A 8				
Block	Task ID	Count	Time	Statistic	Average
1	TaskID:1 (_main)	1	00:00:00.119.317.930	01	00:00:00.119.317.930
1	TaskID:2 (_task1)	58998	00:36:39.231.706.550	248	00:00:00.037.276.370
1	TaskID:3 (_task2)	55327	00:28:02.381.620.730		00:00:00.030.407.960
1	TaskID:4 (_taskx)	217	00:04:58.861.089.710		00:00:01.377.240.040
1	TaskID:5 (_taskx)	211	00:04:50.627.094.680		00:00:01.377.379.590
1	TaskID:6 (_taskx)	227	00:05:12.622.521.670		00:00:01.377.191.720
1	TaskID:7 (_taskx)	187	00:04:17.666.007.190		00:00:01.377.893.080
1	TaskID:8 (_taskx)	196	00:04:27.128.032.600		00:00:01.362.898.120
1	TaskID:9 (_taskx)	200	00:04:35.501.216.990		00:00:01.377.506.080
1	TaskID:10 (_taskx)	245	00:05:37.586.610.540		00:00:01.377.904.530
1	TaskID:11 (_taskx)	230	00:05:16.732.762.290		00:00:01.377.098.960
1	TaskID:12 (_taskx) TaskID:13 (_taskx)		00:05:12.642.146.280 00:05:20.091.103.440		00:00:01.377.278.170 00:00:01.367.910.690
	TaskID:13 (_taskx) TaskID:14 (taskx)		00:05:11.407.982.590		00:00:01.367.910.690
1	TaskID:14 (_Caskx) TaskID:15 (_taskx)		00:04:35.463.363.510		00:00:01.363.680.010
1	TaskID:15 (_taskx) TaskID:16 (_taskx)		00:04:45.062.590.900		00:00:01.377.113.960
1	TaskID:17 (_taskx)	243	00:05:34.681.206.380		00:00:01.377.288.910
1	TaskID:18 (_taskx)		00:04:51.980.619.160		00:00:01.377.267.070
1	TaskID:19 (_taskx)	202	00:04:38.275.539.990		00:00:01.377.601.680
1	TaskID:20 (taskx)	205	00:04:42,425,136,150		00:00:01.377.683.590

Figure 4.5 Realtime Profile Window

How to Open

- Select [View->Performance->Realtime Profile].
- Click the [Realtime Profile] toolbar button.

Description

The two modes of function mode and task mode are available.

This function can acquire profile information for all functions (or tasks) in a maximum of eight block areas, with 128 Kbytes as one block.

Adjacent address areas can be set as blocks.

Task mode:

E100 emulator automatically collects task information when a load module including an OS has been downloaded.

128-Kbyte blocks are automatically assigned to tasks, starting from the lowest task ID.

This function can usually be used without changing any settings.

If there is a task ID that exceeds the block, the block can be put off by clearing the selected state of the task ID with a low number.

Column contents:

Block	Block number
Task ID	Task ID and task entry label name
Count	Number of times the task was called
Time	Accumulated task execution time The display format of the time stamp is as follows: hour:minute:second.millisecond.microsecond.nanosecond
Statistic	Ratio of the Go-Break execution time to the task's Time
Average	Average execution time for one time of execution

When measurement of Count or Time overflows, "overflow" is displayed. In this case, Average is not shown.

Clicking a column header sorts the displayed contents in the ascending order or descending order of that column.

<u> N</u>ote

Up to 8K - 1(= 8191) functions (tasks) can be measured. When the number of functions (tasks) to be measured exceeds 8K - 1(= 8191), that function (task) will not be measured. In this case, that function (task) is grayed.

Popup Menu

The usable options are shown below.

Set Range (S)	Displays the [Set Realtime Profile] dialog box. Either function profile or task profile should be set here (they cannot be measured simultaneously).
Clear All Data (L)	Clears all measured results.
Property (P)	Displays the [Property] dialog box. 10 ns, 20 ns, 40 ns, 80 ns, 160 ns, or 1.6 us can be selected here. This setting will take effect the next time the execution time is measured.
Search (F)	Searches for the specified function. The line where the function was found is highlighted. In task display, the task ID or task entry label is searched for.
Search Next (N)	Searches for the next function/task.
Save In File (V)	Saves all measured results in a file in the text format.

4.2.8 Task Specification for Each Condition

The task ID can be specified in the hardware break, trace, and performance analysis functions. Debugging can be performed while focusing on a specific task by using this specification. For details, refer to the help file of the E100 emulator.

4.2.9 Stack Coverage (Graphical Display of Task Stack Used Size)

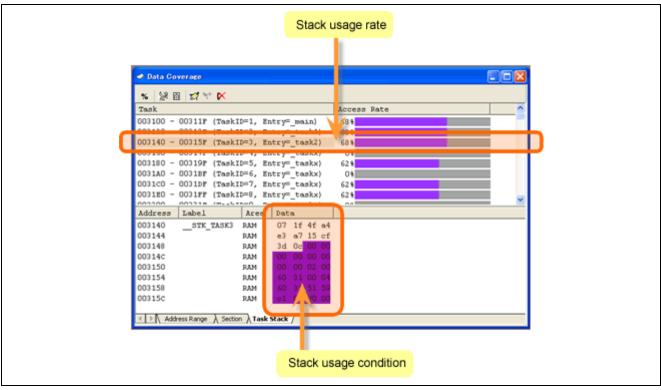


Figure 4.6 Data Coverage Window

How to Open

- Select [View->Code->Data Coverage].
- Click the [Data Coverage] toolbar button.

Description

This window consists of three sheets.

<Address Range> sheet

Data coverage information is collected for the address range specified by the user, and the result is displayed.

<Section> sheet

Data coverage information is collected for the section specified by the user, and the result is displayed.

<Task Stack> sheet

Data coverage information is collected for the task stack specified by the user, and the result is displayed. For this sheet, the window is divided into upper and lower windows by the splitter.

Upper-side Window	
[Task]	Task stack for which coverage information is acquired
[Access Rate]	Displays the percentage and graph of the access rate.
Lower-side Window	
[Address]	Address value
[Label]	Label corresponding to the address (displayed only when a label is set)
[Area]	Memory area This column is blank for areas for which coverage information cannot be acquired.
[Data]	Data value Displayed in bytes. Example: 00 00 FF FF Read/write access is indicated by the background color of the data value.
	Has been accessed: Purple

Popup Menu

The usable options are shown below.

Upper-side window:

Percentage (P)	Calculates and displays the percentage.
Add Range (A)	Adds a new measurement range.
Edit Range (E)	Changes the selected measurement range.
Delete Range (D)	Deletes the selected measurement range.

Lower-side window:

Refresh (R)	Forces a manual update of the displayed contents. Update is performed even when auto update is disabled.
Disable Auto Update (K)	When this is selected, the window is not automatically updated at execution stop. Click this when the window does not have to be updated at each step, such
	as in the case of continuous step execution.
Display Address (A)	Displays the [Display Address] dialog box. A new start address can be input here.
Byte Count Per Line (D)	The number of bytes to be displayed in one line can be specified.
Save (V)	Displays the [Save Coverage Information] dialog box. The location and name of the coverage information file to be saved can be specified here.
Load (L)	Displays the [Load Coverage Information] dialog box. The location and name of the coverage information file to be loaded can be specified here. The only loadable file extension is ".cdv".
Set Hardware (H)	Displays the [Allocate Data Coverage Memory] dialog box. The area in which coverage information is collected should be set here.
Clear Coverage Range (C)	Displays the [Clear Coverage Range] dialog box. The range to be cleared can be specified here.
Clear All Coverage (E)	Clears all data coverage information.

4.2.10 Memory Protection

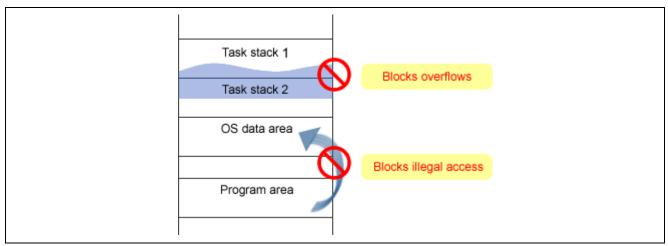


Figure 4.7 Memory Protection

This function detects stack access violation by another task and access violation to the data area of the real-time OS.

Displaying a warning with the balloon of the status bar or setting hardware breakpoints can be done. For details, refer to the help file of the E100 emulator.

4.2.11 Dispatch Detection

This function detects dispatching of a task.

Displaying a warning with the balloon of the status bar or setting hardware breakpoints can be done. For details, refer to the help file of the E100 emulator.

5. Notes

This section describes the notes on the real-time OS aware debugging.

Common Notes for Real-time OS

General	1.	When a task is created or deleted during program execution, the task ID may not be	
	displayed correctly.		
Select OS Definition File dialog box	1.	. When an OS different from the OS in use is selected from the list in the dialog box, operation cannot be performed correctly. In some cases, the debugger does not respond.	
OS Object window		Real-time performance is not achieved while this window is opened because memory accesses will occur (when RAM monitoring is not used).	
	2.	This window may not be displayed correctly in cases, such as, before completion of OS kernel initialization, during program execution, and execution stop in the OS kernel.	
	3.	The maximum number of lines that can be displayed in one sheet is 1023.	
	4.	When using RAM monitoring, this window should be updated to the latest information after the settings have been changed.	
	5.	If there is no valid information, this window may not be displayed correctly (dynamically created task, nonexistent task, nonexistent message, etc.).	
Task Trace window and Task Analyze window		These windows are displayed based on the information of the task that is managed within the OS kernel and currently being executed. Therefore, the timing for task switching or task execution time differs depending on the OS.	
	2.	The time and cycle count are not displayed for some debuggers.	
	3.	When these windows are used, other trace windows or event functions cannot be used in some cases. In turn, when other trace windows or event functions are used, these windows cannot be used in some cases.	
	4.	When these windows are used, real-time performance of the program may be deteriorated. In this case, change the settings in the trace window of each debugger to perform real-time trace or give priority to the CPU, in order to achieve real-time performance. However, note that when real-time performance is achieved, part of the trace information may be lost.	
Task execution history, task execution time, and real-time profile	1.	 These items are displayed based on the information of the task that is managed within the OS kernel and currently being executed. Therefore, the timing for task switching or task execution time differs depending on the OS. 	
Memory protection	ction 1. The OS areas in which illegal access is to be detected differ depending on the OS		
Toolchain	1.	. When the build options of the toolchain (compiler, assembler, etc.) have been changed from those at real-time OS delivery, the real-time OS aware debugging may not operate correctly.	
Emulator debugger	1.	. When memory access during program execution is inhibited by the emulator debugger setting, the task ID of windows are not displayed correctly in some cases.	
E100 emulator debugger	1.	. In step execution that was started by selecting [Debug]->[RTOS Debug], step execution may not halt depending on the user program.	
SH-2A microcomputer	nputer 1. After a CPU reset, if execution stops before OS kernel initialization was completed, the debugger may not respond.		

Notes f	or Individual	Real-time	OS
---------	---------------	------------------	----

HI7200/MP	 The profile ID displayed in Profile in the OS Object window stands for the ID of each task. However note that profile ID 0 indicates the kernel idling state while the maximum profile ID indicates the entire system.
Smalight OS	1. Among the task IDs displayed in each window, 255 means that the OS kernel is idling.
ThreadX	1. The task ID displayed in the debugger is the debugger internal ID and is not the same as the thread ID of the ThreadX.
	2. The maximum number of threads that can be displayed in the OS Object window is 255.
NORTi	1. Among the IDs displayed in the Task Trace window and Task Analyze window, adding 1 to the maximum task ID means that the OS kernel is idling.
osCAN	1. The task ID displayed in the debugger is the debugger internal ID and is not the same as the task ID of the osCAN.
	 When using task stack coverage, click the WithStackCheck check box of the OIL-configurator.
μC/OS-II	1. The task ID displayed in the debugger is the debugger internal ID and is not the same as the task ID of the μ C/OS-II. In addition, the task entry address displayed in the debugger represents the task name.
	2. When the contents of the configuration file (os_cfg.h) are changed from the initial values, the real-time OS aware debugging may not operate correctly.
	3. The maximum number of tasks that can be displayed in the OS Object window is 255.
	 "Ready" shown in Task State in the OS Object window stands for "task is executable", "task is not created", or "task is delayed".
embOS	1. The task ID displayed in the debugger is the debugger internal ID and is not the same as the task ID of the embOS. In addition, the task entry address displayed in the debugger represents the task name.
	2. The maximum number of tasks that can be displayed in the OS Object window is 255.
	3. The task name is not displayed when the XR libraries are used.
μT-Kernel	 Do not change the contents of the system configuration definition file (config¥config.h) from the initial values.
	2. The maximum task ID that can be displayed in the OS Object window is 32.

6. FAQ

No.	Question	Answer
1	The time is not displayed in the Task Trace window and Task Analyze window.	When using the E10A-USB emulator, the time cannot be displayed (except for SH2A-DUAL).
2	The Task Trace window and Task Analyze window are not displayed.	The Task Trace window and Task Analyze window cannot be displayed when using E8a emulator or when using a 14-pin cable with the E10A-USB emulator. When using the E10A-USB emulator, a 36-pin or 38-pin cable should be used.

7. Website and Support

Renesas Technology Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry csc@renesas.com

Revision Record

		Description		
Rev.	Date	Page	Summary	
1.00	Feb.16.10		First edition issued	

- All trademarks and registered trademarks are the property of their respective owners.
- The copyright of the µITRON specification belongs to TRON Association.
- The copyright of the µT-Kernel specification belongs to the T-Engine Forum.
- The copyright of the OSEK/VDX specification belongs to the OSEK/VDX Steering Committee.
- TOPPERS is a registered trademark of the TOPPERS Project.
- OSEK is a registered trademark of Siemens AG.
- ThreadX is a registered trademark of Express Logic, Inc.
- NORTi is a registered trademark of MiSPO Co., Ltd.
- osCAN is a registered trademark of Vector Informatik GmbH.
- Smalight is a registered trademark of Renesas Northern Japan Semiconductor, Inc.
- µC/OS-II is a trademark of Micrium.
- embOS is a trademark of SEGGER Microcontroller.
- All other product names are trademarks or registered trademarks of the respective holders.
- TRON, ITRON, µITRON, and OSEK/VDX are the names of computer specifications and do not indicate a specific group of the commodity or the commodity.

Notes regarding these materials

- 1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
- 2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
- 3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
- 4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
- 5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
- 6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
- 7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
- Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
 (1) artificial life support devices or systems
 - (2) surgical implantations
 - (3) healthcare intervention (e.g., excision, administration of medication, etc.)
 - (4) any other purposes that pose a direct threat to human life

Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.

- 9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
- 10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
- 11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
- 12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
- 13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

© 2010. Renesas Technology Corp., All rights reserved.