#### Old Company Name in Catalogs and Other Documents

On April 1<sup>st</sup>, 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 1<sup>st</sup>, 2010 Renesas Electronics Corporation

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

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

#### Notice

- 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 for written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics atta abooks, 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.
- 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.



# **Debugging Extension**

User's Manual HS6400IWIN1SE Renesas Microcomputer Development Environment System

Renesas Electronics

Rev.1.0 2003.08

## Cautions

Keep safety first in your circuit designs!

 Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- 1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
- 2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- 3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein. The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.

Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).

- 4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- 5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- 6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they
  must be exported under a license from the Japanese government and cannot be imported into a
  country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or

Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.

8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.

The product described in this publication is based on the ITRON specifications and was developed under the guidance of Dr. Ken Sakamura of The University of Tokyo.

- 1. μITRON is an acronym of the "Micro Industrial TRON" and TRON is an acronym of "The Realtime Operating system Nucleus".
- 2. Microsoft® Windows® 98 operating system, Microsoft® Windows® Millennium Edition (Windows® ME) operating system, Microsoft® Windows NT® operating system, Microsoft® Windows® 2000 operating system, and Microsoft® Windows® XP operating system are registered trademarks of Microsoft Corporation in the United States and/or other countries.
- 3. SuperH<sup>™</sup> is a trademark of Renesas Technology Corp.
- 4. All other product names are trademarks or registered trademarks of the respective holders.

# Preface

The Debugging Extension (hereafter called the DX) is software for adding the multitasking debugging functions of the Realtime Operating System (hereafter called the RTOS) to the Highperformance Embedded Workshop (hereafter called the HEW).

This manual describes how the DX is used.

Please read this manual and the related manuals listed below before use so that you fully understand the DX before using it.

This user's manual contains the following four sections and appendixes:

- Section 1 Introduction to the DX
- Section 2 How to operate the DX (based on an example)
- Section 3 The functions of the DX
- Section 4 Precautions and restrictions
- Appendix Troubleshooting

Be sure to read section 4, Precautions and Restrictions, before using the DX.

Please refer to the online help file for details of the DX functions.

The following is a list of the related manuals:

- High-performance Embedded Workshop (HEW) User's Manual
- The user's manual for the RTOS you are using
- The user's manual for the compiler you are using
- The hardware manual and programming manual for the microcomputer you are using

Symbols used in this manual have the following meanings:

The menu name is on the left of the '->', and the item to be selected
from the menu is on the right of the '->'.
(Example: [File -> New]).
The prefix H' is attached for hexadecimal integers. The prefix D' is
attached for decimal integers. If no prefix is attached, a decimal integer
is assumed.

# Contents

1.1       Outline       1         1.2       Features       2         1.3       Preparations for Using the DX       2         Section 2       Tutorial       3         2.1       Configuring the Tutorial       3         2.1       Configurator       3         2.1.1       Configurator       3         2.1.2       Build       3         2.2       Contents of the Sample Program       4         2.3       Executing the Sample Program       5         Section 3       Functions       2         3.1       Referencing the States of Objects       22         3.1       Referencing the States of Objects       22         3.2       Manipulating the Objects       24         3.3       Displaying Service Call Trace       22         3.4       List of Functions       24         3.4.1       Menu Items       24         3.4.2       Windows and Dialog Boxes       36         Section 4       Precautions and Restrictions       36
1.3       Preparations for Using the DX.       2         Section 2       Tutorial
Section 2Tutorial32.1Configuring the Tutorial32.1.1Configurator32.1.2Build32.2Contents of the Sample Program42.3Executing the Sample Program5Section 3Functions23.1Referencing the States of Objects23.2Manipulating the Objects23.3Displaying Service Call Trace23.4List of Functions23.4.1Menu Items23.4.2Windows and Dialog Boxes3Section 4Precautions and Restrictions3
2.1       Configuring the Tutorial       3         2.1.1       Configurator       3         2.1.2       Build       3         2.2       Contents of the Sample Program       4         2.3       Executing the Sample Program       5         Section 3       Functions       2         3.1       Referencing the States of Objects       2         3.2       Manipulating the Objects       2         3.3       Displaying Service Call Trace       2         3.4.1       Menu Items       2         3.4.2       Windows and Dialog Boxes       3         Section 4       Precautions and Restrictions       3
2.1.1Configurator
2.1.2       Build       3         2.2       Contents of the Sample Program.       4         2.3       Executing the Sample Program.       5         Section 3       Functions       2         3.1       Referencing the States of Objects       2         3.2       Manipulating the Objects       2         3.3       Displaying Service Call Trace       2         3.4       List of Functions       2         3.4.1       Menu Items       2         3.4.2       Windows and Dialog Boxes       3         Section 4       Precautions and Restrictions       3
2.2       Contents of the Sample Program
2.3 Executing the Sample Program5Section 3 Functions23.1 Referencing the States of Objects23.2 Manipulating the Objects23.3 Displaying Service Call Trace23.4 List of Functions23.4.1 Menu Items23.4.2 Windows and Dialog Boxes3Section 4 Precautions and Restrictions3
2.3 Executing the Sample Program5Section 3 Functions23.1 Referencing the States of Objects23.2 Manipulating the Objects23.3 Displaying Service Call Trace23.4 List of Functions23.4.1 Menu Items23.4.2 Windows and Dialog Boxes3Section 4 Precautions and Restrictions3
3.1       Referencing the States of Objects       22         3.2       Manipulating the Objects       26         3.3       Displaying Service Call Trace       26         3.4       List of Functions       29         3.4.1       Menu Items       29         3.4.2       Windows and Dialog Boxes       30         Section 4       Precautions and Restrictions       31
3.1       Referencing the States of Objects       22         3.2       Manipulating the Objects       26         3.3       Displaying Service Call Trace       26         3.4       List of Functions       29         3.4.1       Menu Items       29         3.4.2       Windows and Dialog Boxes       30         Section 4       Precautions and Restrictions       31
3.2       Manipulating the Objects       20         3.3       Displaying Service Call Trace       21         3.4       List of Functions       22         3.4.1       Menu Items       22         3.4.2       Windows and Dialog Boxes       30         Section 4       Precautions and Restrictions       31
3.3       Displaying Service Call Trace       24         3.4       List of Functions       29         3.4.1       Menu Items       29         3.4.2       Windows and Dialog Boxes       30         Section 4       Precautions and Restrictions       31
3.4       List of Functions       29         3.4.1       Menu Items       29         3.4.2       Windows and Dialog Boxes       30         Section 4       Precautions and Restrictions       30
3.4.2       Windows and Dialog Boxes
Section 4 Precautions and Restrictions
4.1 Realtime Operation of the User System and Cyclic Handler
4.2 Consistency of Displayed Contents
Appendix A Troubleshooting
A.1 Displaying [Action Result] Window Results

# Section 1 Overview

## 1.1 Outline

The DX is software for debugging applications which are created for the RTOS. It is installed in the HEW and RTOS systems.

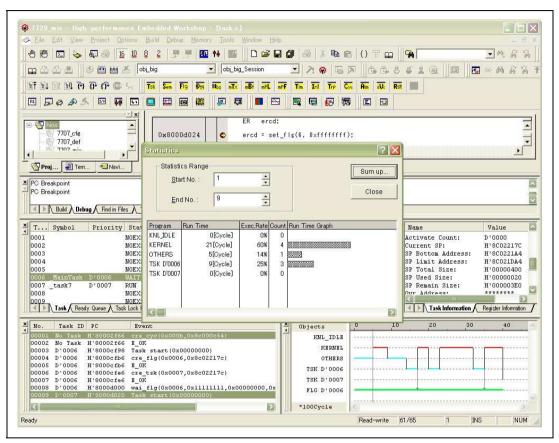


Figure 1.1 DX Overview

#### **1.2 Features**

• Graphical user interface

The status of objects such as tasks can be referred to and modified through windows and dialog boxes, and multitasking applications can be debugged in the HEW environment.

• Display of tracing information for service calls

The history of service calls in RTOS systems can be displayed graphically. The user can also select the items to be displayed in this history.

## **1.3** Preparations for Using the DX

The followings must be done before using the DX for debugging.

Install the DX: Follow the instructions by the installer to install the DX.

**Configure the RTOS:** Configure the RTOS system you are using to match the DX. For an example of configuration, refer to section 2, Tutorial.

**Initiate the HEW:** Initiate the HEW so that multitasking debugging is enabled after a program has been built. For examples of the use of the DX, refer to section 2, Tutorial. For the functions of the DX, refer to section 3, Functions, and to the online help file.

# Section 2 Tutorial

Examples of the operation of the DX using the HI7000/4 series as the RTOS are explained in this section. For details, refer to the online help file.

The following environment is assumed.

- RTOS: HI7700/4
- Microcomputer: SH7729
- Debugger: SuperH<sup>TM</sup> RISC engine simulator/debugger

## 2.1 Configuring the Tutorial

The sample provided by the HI7700/4 is used. For details on the configuration, refer to section 5, Configuration, in the HI7000/4 Series User's Manual.

#### 2.1.1 Configurator

The sample HCF file (hiuser\sh7729\7729.hcf) provided by the HI7700/4 is used to create the configuration file. The following configurator settings must be made.

- Check CFG\_ACTION.
- Check CFG\_TRACE.
- Set CFG\_TRCTYPE to 'emulator trace'. (This setting allows RTOS tracing by the DX even in the simulator.)
- Set CFG\_MAXTSKID to seven or more.
- Set CFG\_STSTKID to five or less
- Set CFG\_MAXTSKPRI to seven or more.
- Set CFG\_MAXFLGID to six or more.
- Install the following service calls: cre\_tsk, ext\_tsk, cre\_flg, wai\_flg, and set\_flg

#### 2.1.2 Build

The whole linkage method of HI7700/4 is used in this example. Use the HEW to open the sample hios.hws, and select 7729\_mix as the project name for the build. Select [Options->SuperH RISC engine Standard Toolchain...] and then [User] from [Gbr relative logic operation] in the [Optimize] category on the [C/C++] page. Select [Build->Build] to start a build.

### 2.2 Contents of the Sample Program

The sample source program is hiuser\tutorial\task.c, which includes two tasks, MainTask and Task7.

When the system is initiated, MainTask is automatically executed because creation and initiation of MainTask is specified by the configurator with the following conditions:

- Task ID: 6
- Initial priority: 6
- Start address: MainTask()
- Initial state: READY (the TA\_ACT attribute is specified)

TASK7 is dynamically created by MainTask with the following conditions:

- Task ID: 7
- Initial priority level: 7
- Start address: Task7()
- Initial state: READY (TA\_ACT attribute is specified)

The priority of MainTask is higher.

MainTask creates an event flag with ID6 and Task7, and uses the service call wai\_flg to wait for the event flag with ID6. Then, the event flag with ID6 is deleted with the service call del\_flg, and MainTask is exited by the service call ext\_tsk.

Task7 uses the service call set\_flg to set the event flag with ID6, then enters an endless loop with a while(1); statement.

Note that this sample program is created for the user to learn the operation of the kernel and the DX, and its contents in terms of processing have no meaning.

### 2.3 Executing the Sample Program

(1) Start up the simulator/debugger. Select [Options->Debug Settings...] in the HEW to display the [Debug Settings] dialog box. Then select 7729\_mix from the list on the left, [SH3-DSP Simulator] from the [Target] list box on the [Target] page, and hiuser\obj\_big\7729\_mix.abs of HI7000/4 by the [Add...] button, respectively.

bj_big_Session	Target Options	
	SH3-DSP Simulator	>
7707_mix	Default Debug Format.	
	Elf/Dwarf2	
	Download Modules: Offset Ad Format	Add
	ser¥obj_big¥7729_mix.abs H'00000000 Elf/Dwarf2	$\sim$
		Remove
		Modify
		Up
7729 def		Down

Figure 2.1 [Debug Settings] Dialog Box

(2) Start up the DX. Select [RTOS HI7000/4\_4DX ECX] from [Project -> Components...]. Then press [Load] and [OK].

Project: 7729_mix	<b>•</b>
Application Extensions	
	Load

Figure 2.2 [Component Gallery] Dialog Box

(3) Set up the simulator/debugger. Check [Enable Timer] in the [Simulator System] dialog box displayed by selecting [Options -> Simulator -> System...], and select [Continue] from the [Execution Mode] list box.

SH3-DSP		•
<u>B</u> it size:  D'32	System Call Address:	
<u>E</u> ndian: Big Endian	Execution Mode:	7
Response: D'40000	Round Mode: Round To Zero	•
Clock Rate:	Step Unit:	
 Peripheral Clock Rate: 🛛 🔽 Enable ]	Stage	-

Figure 2.3 [Simulator System] Dialog Box ([System] Page)

(4) The memory resources listed in table 2.1 must be ensured in the [Memory] page of the [Simulator System] dialog box.

Start Address	End Address	Attribute
H'00000000	H'0000FFFF	Read
H'0C000000	H'0C0FFFFF	Read/Write

#### Table 2.1 Memory Resources for the Simulator/Debugger

	p:	¢_		_		Memory <u>R</u> es	End	
Begin	End	Туре	S	S	-	Begin	- Laurence	Attribute
00000000	03FFFFFF	EXT	32	1	<	00000000	0000FFFF	Read
04000000	04FFFFFF	1/0	32	1		05007000	05000FFF	Read/Write
05007000	05008FFF	XRAM	32	1	100	05017000	05018FFF	Read/Write
05017000	05018FFF	YRAM	32 32 32	2	<	0000000	OCOFFFFF	Read/Write
08000000 E0000000	DFFFFFFF	EXT I/O	32	1				
20000000	TELEFTER	10	32	100				
						1		

Figure 2.4 [Simulator System] Dialog Box ([Memory] Page)

(5) Download the load module. Select hiuser\obj\_big\7729\_mix.abs from [Debug->Download Modules].

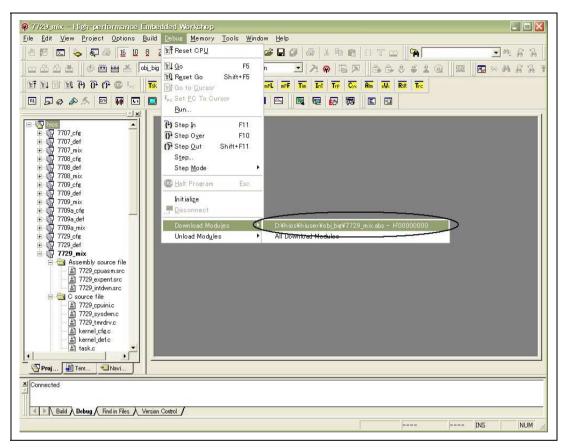


Figure 2.5 Downloading the Load Module

(6) Make the settings for RTOS tracing. Select [View->RTOS->Trace]. Click the right-hand mouse button on the [RTOS Trace] window and then select [Set Trace Options...] from the popup menu.

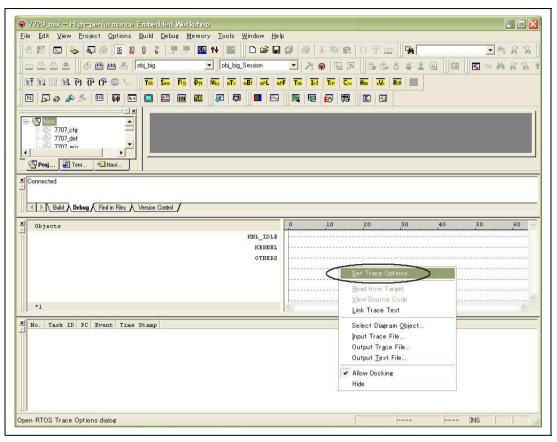


Figure 2.6 Setting RTOS Tracing

(7) Select [Cycle] from [Scale Type] and [\*100 Cycle] from [Scale List], respectively.

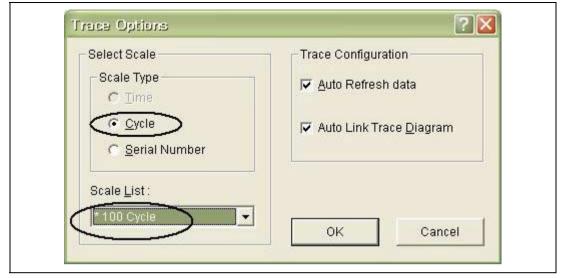


Figure 2.7 [Trace Options] Dialog Box

(8) Select [Debug->Reset Go] to execute the program.

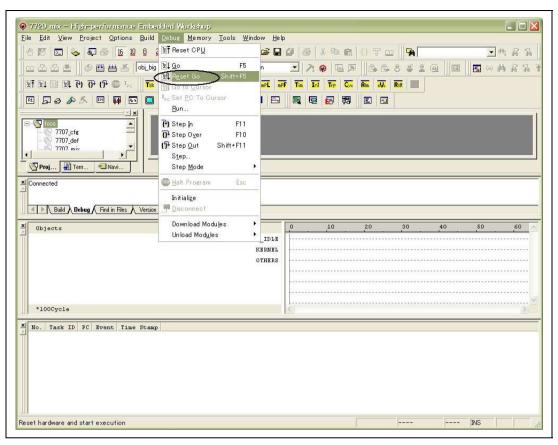


Figure 2.8 Executing the Program

(9) Select [Debug->Halt Program] to stop the program execution.

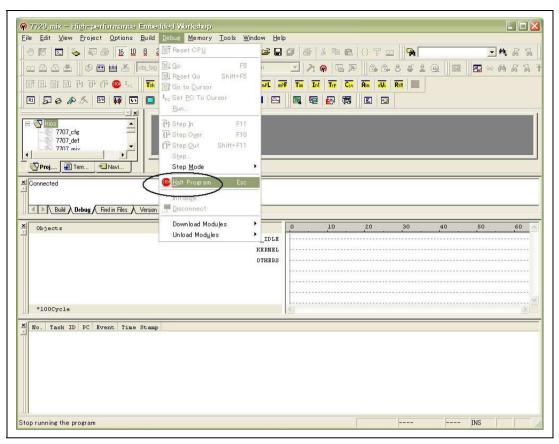


Figure 2.9 Stopping the Program Execution

(10) Execution stops within the while(1); statement of Task7 as shown in figure 2.10.

17199 mit	ize – Elfaker	าวกับการก	ce Embedded Workshop - [task.c]						
> File Edi		Project Op	and the second						
	it <u>v</u> iew <u>r</u>	Project Op				10			- 8 >
	2 🕹 🗟	• - 11-			X ⓑ €   {} T œ	- 111	- 10 1	<b>→</b> #	
	i≞∥ø		obi_big_Se	ession 💌 🤇			<u>)</u>		183
et et et	i II 7) 0	ት 🕼 🗘 ተ	c Tsk Sem F19 Dt9 Mbx mTx mb	Bf mPL mPF Tim	Int Trp Cyc Aim SVc	Rst			
RI 💭 🤅	a 📣 🍝	10							
- 67	7707_cfg 7707_def 7707_miv	-		rcd = set_flg(6, 0 nile(1);	Dxffffffff);				•
Connected		4-							
Stop									
		Q	7						
	Build A Debug ,	Find in Files	IV: OIL						
A Research Street Press		I THIN IT THES	V version Control						
		1 1 more 1 1 more	V version Control V		10 20	30	40	50	60
Object	:5				10 20	30	. <sup>40</sup>	50	60
Object	:5			KNL_IDLE	10 20	,30	40	,50	60
Object	;5	A Consense conse		KNL_IDLE	,10 ,20	30	.40 	50	,60 
Object	15			KNL_IDLE	10 ,20	,30	40	50	<u>60</u>
Object	25	C 1110 11 1 1105		KNL_IDLE	10 20	30	.40 	,50	<u>60</u>
Object	25			KNL_IDLE	10 20	30	40	50	<u>,60</u>
Object	25			KNL_IDLE	10 20	30	<u>,40</u>	50	<u>60</u>
				KNL_IDLE	10 20	30	<u>40</u>	50	,60
Object: *100Cy				KNL_IDLE	10 20	30	40	50	60
*100Cy	7cle		- <u> </u>	KNL_IDLE		30	40	50	60
*100Cy	ycle Task ID P	PC	Fvent	KNL_IDLE	Time Stamp	30	40	50	60
*100Cy	rcle Task ID P To Task H'	PC '80002166	Event cre_cyc(0x000b,0x8c000c54)	KNL_IDLE	Time Stamp 0000015828[Cycle]	,30	40	<u>,</u> 50	60
*100Cy *100Cy 00001 No 00002 No	rcle Task ID P To Task H' O Task H'	PC '80002166 '80002166	Event cre_cyc(0x000b,0x8c000c54) g_0K	KNL_IDLE	Time Stamp 0000015628[Cycle] 0000015628(Sycle]	30	40	50	60
*100Cy *100Cy No. 1 00001 No 00002 No	rcle Task ID P To Task H' Task H' '0006 H'	PC '80002166 '80002166 '80002168	Event re_cyc(0x000b,0x8c000c54) E_OR Task start(0x00000000)	KNL_IDLE	Time Stamp 0000015828[Cycle] 0000016263[Cycle] 0000016276[Cycle]	30	40	50	<u>,60</u>
*100Cy *100Cy 00001 No 00002 No 00002 No	Task ID P To Task H' To Task H' To Task H' '0006 H'	PC '80002166 '80002166 '8000c198 '8000c196	Event cre_cyc(0x000b,0x8c000c54) g_0R Task start(0x00000000) cre_f14(0x0006,0x8c02217c)	KNL_IDLE	Time Stamp 0000015628[Cycle] 0000015728]Cycle] 0000016778[Cycle] 000001778[Cycle]	30	40	50	<u>,60</u>
*100Cy *100Cy 00001 No 00002 No 00002 D 00004 D 00005 D	701e Task ID P To Task H' 10006 H' 10006 H'	PC *80002166 *80002166 *80002169 *80002169 *80002169	Event Event Cre_cyc(0x000b,0x8c000c54) E_0K Task start(0x0006,0x8c02217c) K_0K	KNL_IDLE	Time Stamp 0000015828[Cycle] 0000016252[Cycle] 0000017051[Cycle] 0000017051[Cycle] 0000017052[Cycle]	20	40	50	<u></u>
*1000Cy *1000Cy 00001 No 00002 No 00004 D 00004 D 00005 D 00006 D	701e Task ID P 10 Task H' 10006 H' 10006 H' 10006 H'	PC *80002166 *80002166 *8000c1b6 *8000c1b6 *8000c1b6	Event           cre_cyc(0x000b,0x8c000c54)           B_0K           Task start(0x00000000)           cre_f1g(0x0006_0x8c02217c)           B_0K           Te_tosk(0x0007,0x8c02217c)	KNL_IDLE	Time Stamp 0000015828[Cycle] 0000016263[Cycle] 000001773[Cycle] 0000017351[Cycle] 000001732[Cycle] 0000017365[Cycle]	20	40	50	
*100Cy *100Cy 00001 NG 00002 NG 00005 D 00005 D 00005 D	rcle Task ID F fo Task H' '0006 H' '0006 H' '0006 H'	PC *80002166 *80002166 *80002166 *80002166 *80002166 *80002166	Event Ev	KNL_IDLE KEENEL OTHERS	Time Stamp 000015828[Cycle] 0000016735[Cycle] 0000016778[Cycle] 0000017732[Cycle] 0000017765[Cycle] 0000017651[Cycle]	30	40	50	<u>,60</u>
*1000cy No. 1 00001 No 00002 No 00003 D 00005 D 00005 D 00005 D 00005 D	rcle Task ID P To Task H' '0006 H' '0006 H' '0006 H' '0006 H'	PC 180002166 180002166 1800021b6 1800021b6 1800021b6 1800021c6 1800021c6 1800021c6 1800021c6 180002100 180000100 18000000000 180000000000	Event           cre_cyc(0x000b,0x8c000c54)           E_OK           Task start(0x00000000)           cre_f1g(0x0006,0x8c02217c)           E_OK           Zeve_tsk(0x0007,0x8c02217c)           E_OK           Mai_f1g(0x0006,0x1111111,0x0000	KNL_IDLE KEENEL OTHERS	Time Stamp 0000015828[Cycle] 0000016263[Cycle] 0000017051[Cycle] 0000017051[Cycle] 0000017051[Cycle] 000001766[Cycle] 0000018531[Cycle] 0000018531[Cycle]	30	40	50	<u>60</u>
*100Cy *100Cy 00001 No 00002 No 00003 D 00005 D 00006 D 00006 D 00006 D	Task ID P Task H' '0006 H' '0006 H' '0006 H' '0006 H'	PC *80002166 *80002166 *80002198 *80002198 *80002186 *80002186 *80002186 *80002186 *80002186	Event Event Cre_cryc(0x000b,0x8c000c54) E_0R Task start(0x00000000) cre_f1g(0x0006,0x8c02217c) E_0R g_0R wai_f1g(0x0006,0x1111111,0x0000 Task start(0x0000000)	KNL_IDLE KEENEL OTHERS	Time Stamp           0000015828 [Cycle]           0000016738 [Cycle]           0000016778 [Cycle]           0000017372 [Cycle]           0000017365 [Cycle]           0000017365 [Cycle]           0000017365 [Cycle]           00000107665 [Cycle]           000001805 [Cycle]           000001805 [Cycle]           000001800 [Cycle]	30	40	50	<u>60</u>
*100Cy *100Cy 00001 No 00002 No 00005 D 00005 D 00006 D 00007 D 00007 D 00007 D	Task ID P fo Task H' '0006 H' '0006 H' '0006 H' '0006 H' '0006 H' '0006 H'	PC 180002166 180002166 180002166 180002166 180002166 180002166 180004168 180004018 180004000 1800040020	Event           cre_cryc(0x000b,0x8c000c54)           B_0K           Task start(0x00000000)           cre_tsk(0x0007,0x8c02217c)           B_0K           Waif f1g(0x006,0x1111111,0x0000           Task start(0x0000000)           sae_tf1g(0x006,0x1111111,0x000	KNL_IDLE KEENEL OTHERS	Time Stamp           0000015828 [Cycle]           0000016263 [Cycle]           0000017051 [Cycle]           0000017051 [Cycle]           0000017053 [Cycle]           0000017053 [Cycle]           0000018531 [Cycle]           0000018531 [Cycle]           0000019400 [Cycle]           0000019400 [Cycle]	30	40	50	60
*100Cy *100Cy 1 No. 1 00001 No 00003 D 00003 D 00005 D 00006 D 00006 D 00008 D	Task ID P fo Task H' '0006 H' '0006 H' '0006 H' '0006 H' '0006 H' '0006 H'	PC *80002166 *80002166 *80002198 *80002198 *80002186 *80002186 *80002186 *80002186 *80002186	Event           cre_cryc(0x000b,0x8c000c54)           B_0K           Task start(0x00000000)           cre_tsk(0x0007,0x8c02217c)           B_0K           Waif f1g(0x006,0x1111111,0x0000           Task start(0x0000000)           sae_tf1g(0x006,0x1111111,0x000	KNL_IDLE KEENEL OTHERS	Time Stamp           0000015828 [Cycle]           0000016738 [Cycle]           0000016778 [Cycle]           0000017372 [Cycle]           0000017365 [Cycle]           0000017365 [Cycle]           0000017365 [Cycle]           00000107665 [Cycle]           000001805 [Cycle]           000001805 [Cycle]           000001800 [Cycle]	30	40	50	

Figure 2.10 Source Window with the Execution Stopped

(11) You can check how the program was executed by using the service-call trace function. To display MainTask (task ID 6) and Task7 (task ID 7) in such a way that the diagram is easy to see, click the right-hand mouse button on the [RTOS Trace] window and select [Select Diagram Object...] from the pop-up menu. Add task IDs 6 and 7, and close the dialog box.

Available Object Item : Task EventFlag Semaphore Mailbox MessageBuffer	S <u>e</u> t:	Selected Object Item : KNL_IDLE KERNEL OTHERS TSK D'0006 TSK D'0007
V-MemoryPool F-MemoryPool Interrupt DataQueue Mutex	Delete :	
D' 6 - D' 7		OK Cancel

Figure 2.11 [Select Diagram Object] Window

(12) Task IDs 6 and 7 are displayed in the [RTOS Trace Diagram] window. With a glance at the window, you can see that the execution of Task7 starts after MainTask has issued wai\_flg. Double-clicking the row where the service call is displayed in the [RTOS Trace Text] window opens the corresponding source window.

A reason to the second se	
7729_mix = High-performance Embedded Workshop = [task.c]	
<u>File Edit View Project Options Build Debug Memory Tools Window</u>	Help _ Ə ×
🕀 🕅 🔽 🗞 🖓 🍜 16 12 8 2 🦉 🥂 🌆 🙌 🌃 🗅 🖨 🖨	0 5 % P 2 {} T 2 9 7 4 7 3 1
🖸 🏠 🏝 🕼 📇 👗 🚺 obi_big 🔽 🔽 🔽 💽	- X 🛊 🗳 X 🕼 🖧 8 2 2 0 🗰 🖬 🗠 44 6 7 3 1
ET EL ET EL ?) (? @ Iec Tsk Sem Fig Dtg Mbx mTx mBf mPL mP	F Tim Int Trp Cyc Am sVc Rat
	R R 😥 🕅 E 🖸
× ×	
- 0x8000cfe8 waiptn = 0x	or eventflag=6 ***/
Proj Tem Navi	
Dbjects KNL_IDLE KFFMEL OTHERS TSK D'0006	
TSK D'0007	
TSK D'0007	
*100Cycle	
*100Cycle X No. Task ID PC Event	
+100Cycle	Time Stamp         Image: Stamp
*100Cycle M No. Task ID PC Event 00001 No Task H'80002f66 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002f66 E_OR 00003 D'0006 H'80002f86 Task start(0x00000000)	0000015828[Cycle]
*100Cycle No. Task ID PC Event 00001 No Task H'80002f66 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002f66 E_OR 00003 D'0066 H'8000cf66 Task start(0x00000000) 0004 D'0066 H'8000cf66 cre_lig(0x0006,0x8c02217c)	0000015828[Cycle] 0000016263[Cycle] 0000016778[Cycle] 0000017051[Cycle]
*100Cycle X No. Task ID PC Event 1 0001 No Task H'80002f66 Cre_cyc(0x000b,0x8c000c54) 0002 No Task H'80002f66 E OK 00003 D'0006 H'8000cf68 Task start(0x00000000) 00004 D'0006 H'8000cf66 Cre_f1g(0x0006,0x8c02217c) 00005 D'0006 H'8000cf66 Cre_f1g(0x0006,0x8c02217c) 00005 D'0006 H'8000cf66 Cre_f1g(0x0006,0x8c02217c) 00005 D'0006 H'8000cf66 Cre_f1g(0x0006,0x8c02217c)	0000015828[Cycle] 00000167283[Cycle] 0000016778[Cycle]
*100Cycle No. Task ID PC Event 00001 No Task H'80002f66 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002f66 E_OK 00003 D'0006 H'8000cf68 Task start(0x00000000) 00004 D'0006 H'8000cf66 cre_f1g(0x0006,0x8c02217c) 00005 D'0006 H'8000cf66 cre_tsk(0x0007,0x8c02217c)	0000015828[Cycle] 0000016263[Cycle] 0000016778[Cycle] 0000017051[Cycle] 0000017372[Cycle] 0000017656[Cycle]
*100Cycle  X No. Task ID PC Event  00001 No Task H'80002f66 Cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002f66 E OR 00003 D'0006 H'8000cf98 Task start(0x00000000) 00004 D'0006 H'8000cf96 Cre_f10(0x0006,0x8c02217c) 00005 D'0006 H'8000cf66 E OR 00006 D'0006 H'8000cf66 Cre_tsk(0x0007,0x8c02217c) 0007 D'0006 H'8000cf66 Cre_tsk(0x007,0x8c02217c) 0007 D'0006 H'8000cf66 Cre_tsk(0x007,0x8c02217c) 0007 D'000 H'8000cf66 Cre_tsk(0x007,0x8c02217c) 0000 D'000 D'000 H'8000cf66 Cre_tsk(0x000000000000 H'8000000000000000000000000	0000015828[Cycle] 0000016263[Cycle] 0000016778[Cycle] 0000017051[Cycle] 000001705[Cycle] 0000017366[Cycle] 0000017666[Cycle]
*100Cycle X No. Task ID PC Event 00001 No Task H'80002f66 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002f66 J.0K 00003 D'0006 H'8000cf66 J.0K 00005 D'0006 H'8000cf66 cre_f1g(0x0006,0x8c02217c) 00005 D'0006 H'8000cf66 G.0K 00006 D'0006 H'8000cf66 K 0K 00006 H'8000cf66 K 0K 00006 H'8000cf66 K 0K	0000015828[Cycle] 0000016785(Cycle] 0000017372[Cycle] 0000017372[Cycle] 0000017366[Cycle] 0000017666[Cycle] 0000017665[Cycle] 00000188051[Cycle]
*100Cycle  X No. Task ID PC Event 00001 No Task H'80002166 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002166 E_OR 00003 D'0006 H'8000c168 Task start(0x0000000) 00004 D'0006 H'8000c166 E_OR 00005 D'0006 H'8000c166 E_OR 00005 D'0006 H'8000c166 E_OR 00005 D'0006 H'8000c166 I_OR 00005 D'0007 H'80000000 T_AS tart(1g(D*00000000) OXEC	0000015828[Cycle] 0000016263[Cycle] 0000017051[Cycle] 0000017782[Cycle] 00000177851[Cycle] 0000017666[Cycle] 0000018605[Cycle] 00201940[Cycle] 0000019400[Cycle]
*100Cycle X No. Task ID PC Event 00001 No Task H'80002166 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002166 K_0K 00003 D'0006 H'8000c168 Task start(0x0000000) 00004 D'0006 H'8000c166 E_0K 00006 D'0006 H'8000c166 E_0K 00006 D'0006 H'8000c166 E_0K 00006 D'0006 H'8000c166 K_0K 00007 D'0006 H'8000c166 S_0K 00009 D'0007 H'8000c166 S_0K 00009 D'0007 H'8000c160 Vail111111,0x00000000,0x8c 00009 D'0007 H'8000c103 Sec Ligt(0x000,0x1111111,0x00000000,0x8c 00009 D'0007 H'8000c103 Sec Ligt(0x000,0x1111111,0x00000000,0x8c	0000015828[Cycle] 0000016778[Cycle] 0000017372[Cycle] 0000017372[Cycle] 0000017372[Cycle] 0000017866[Cycle] 0000018503[Cycle] 0000019681[Cycle] 0000019681[Cycle] 0000019681[Cycle]
*100Cycle  X No. Task ID PC Event 00001 No Task H'80002166 cre_cyc(0x000b,0x8c000c54) 00002 No Task H'80002166 E_OR 00003 D'0006 H'8000c168 Task start(0x00000000) 00004 D'0006 H'8000c166 E_OR 00005 D'0006 H'8000c166 E_OR 00005 D'0006 H'8000c166 E_OR 00006 D'0006 H'8000c166 E_OR 00008 D'0007 H'8000c168 Start(0x0000000) 0008 D'0007 H'8000c168 Start(0x0000000) 0008 D'0007 H'8000c16 Start(0x000000)	0000015828[Cycle] 0000016263[Cycle] 0000017051[Cycle] 0000017782[Cycle] 00000177851[Cycle] 0000017666[Cycle] 0000018605[Cycle] 00201940[Cycle] 0000019400[Cycle]

Figure 2.12 [RTOS Trace] Window

(13) You can check the states of MainTask and Task7 in the [Task] window. Select [View ->RTOS->Task] to open the [Task] window. The MainTask is in the DORMANT state, while Task7 is in the RUN state.

6									
the second s	mix — Fligh-performan	and the second	A second s						
I Eile E	<u>Edit V</u> iew <u>P</u> roject <u>O</u> p	tions <u>B</u> uild <u>D</u> ebug <u>I</u>	<u>M</u> emory <u>T</u> ools <u>W</u> ind	low <u>H</u> elp					- 8 ×
	🖸 🌭 🖗 🚳 🚹	10 8 2 🦉 🏴 🛛	🔂 👬 🔛 🔛 🖸 🖬		3 <b>B B</b>	{}T 🛱			• M & A
	£ ≝ ∭ @ ∰ <b>∺</b> ⊀	obi_big	▼ obi_big_Session	- 7	•			0    III    III    III	» M A A T
I IT IL	🗄 🖪 🖰 🗗 🗗 🗐	er Tsk Sem Flg D	g Mbx mTx mBf mb	L mPF Tim	Int Trp Cyc	Alm SVc	Rstt		
		Version Control /			2 🕢 🕅				
× Obje	cts			0	10	20	30	40 50	60 📈
			KNL 1	DLE					
			KEI	NEL		r			
			OTH	IERS					
			TSK D'O	1006		-	2007 <b>- 1</b> 000		
			TSK D'O	1007					
									· · · · · · · · · · · · · · · · · · ·
*100	Cycle			<					2
× No.	Task ID PC	Event			Time Stamp				
		cre_cyc(0x000b,0x8c	000c54)		0000015828[				
		E_OK Task start(0x000000	00)		0000016263[				
		cre flg(0x0006,0x8c			0000017051[				
		E_OK			0000017372[				
		cre_tsk(0x0007,0x8c E OK	02217c)		0000017666[				<u> </u>
		wai_flg(0x0006,0x11	111111,0x00000000	0x8c02219c)	0000018805[				
		Task start(0x000000			0000019400[				
00010		<pre>set_flg(0x0006,0xff R OK</pre>	ffffff)		0000019664[				
× Tsk II	D Symbo	ol Fr	t.Inf	Priority		ate	1	Wait Factor	Attr:
0001	o Samo	D1 2X	CTUL	FLIGLICY	NOF			Wart Factor	AUCL.
0002					NOI				
0003					NOI				
0004					NO				
0005	Main	Toob HIO	0000000	D'0006	NOI	EXS RMANT			TA CO:
0000	_hain _task'		0000000	D'0007	C RUI		5		IA_00.
0008				N 9989	MOR			3)	
	Task & Ready Queue & Task L	ock Muter						1	
	I HAR V LIGHTA CHERE V 192K	LUCK INIMER J					Provide State		-
Ready						Read-write	43/65	1 INS	NUM //

Figure 2.13 [Task] Window

(14) Next, execute the user program when the priority of MainTask is lower than Task7. First, MainTask and Task7 must be deleted to return them to their initial state. To delete MainTask, select (highlight) MainTask in the [Task] window, click the right-hand mouse button to launch the pop-up menu, then select [Action->Delete Task...] from this pop-up menu.

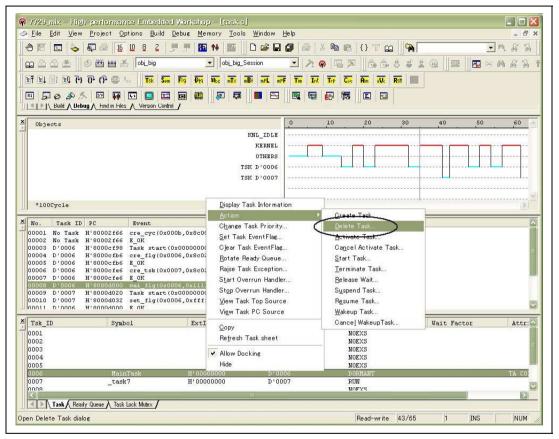


Figure 2.14 [Task] Window Pop-Up Menu

(15) Press the [OK] button in the [Delete Task] dialog box.

Figure 2.15 [Delete Task] Dialog Box

(16) Since Task7 is still in the RUN state, it must be forcibly terminated before deletion. To terminate Task7, select (highlight) Task7 in the [Task] window, click the right-hand mouse button to launch the pop-up menu, then select [Action->Terminate Task...]. This opens the [Terminate Task] dialog box, so press the [OK] button. Task7 can then be deleted in the same way as MainTask.

Tsk_ID: D'  7	
Force Terminate	Cancel

Figure 2.16 [Terminate Task] Dialog Box

(17) You can check the result of the manipulation of kernel objects, such as the termination and deletion of a task, by selecting [View->RTOS->Action Result]. However, even if the [Action Result] window is opened, the results will not be visible here. The DX sends requests for the execution of service calls to the kernel to realize manipulation of kernel objects. Thus a task will be actually deleted after execution by the kernel. To obtain the results in this case, resume the execution of the target by selecting [Debug->Go]. The results will be automatically displayed. Then select [Refresh task sheet] from the pop-up menu of the [Task] window. You will see that MainTask (task ID 6) and Task7 (task ID 7) are both in the NOEXS state.

🛞 7729 mix - High-ge	erformance Embedded Wo	rkshon – Frask el				
		g Memory Tools Win	dow Help			_ 8 ×
	<u>∰ 16 10 8 2 9 9</u>					• M & %
🖸 🕰 🖾 🏛 🖉 🕯	🛗 📇 🚺 obi_big	Job_big_Session	- × 🖗		8 2 2 9 1 9 1 9	× M A A 1
TO PL OF DI PL OF		Dtg Mbx mTx mBf m	PL mPF Tim Int Tri	Cyc Aim sVc	Rat	
RI 5 6 5 5						
III .IOOCYCIE						10
× No. Task ID PC	Event		Time	Stamp		
	0002f66 cre_cyc(0x000b,	0x8c000c54)		15827[Cycle]		
00002 No Task H'8 00003 D'0006 H'8	0002166 E_0K 000cf98 Task start(0x00	000000)		16262[Cycle] 16777[Cycle]		-
	000cfb6 cre_flg(0x0006,	0x8c02217c)		17050[Cycle]		1 <b>.</b>
	000cfb6 E_0K 000cfe6 cre tsk(0x0007,	0x8c02217c)		17371[Cycle] 17665[Cycle]		
	000cfe6 E_0K			18530[Cycle]		_
	000d000 wai_flg(0x0006, 000d020 Task start(0x00			18804[Cycle] 19399[Cycle]		
	000d032 set_flg(0x0006, 000d000 R OK	Oxfffffff)		19663[Cycle] 20155[Cycle]		
	anniannin k nk			2015510we1e1		
Tsk_ID	Symbol	ExtInf	Priority	State	Wait Factor	Attr:
0001				NOEXS NOEXS		
0003				NOEXS		
0004				NOEXS		
0005			/	NOEXS		
0007			(	NOEXS		
0008		0.m		2 2 1118		
Task Ready Que	eue 👌 Task Lock Mutex 🖊					
× No.	SVC	Ercd	ReturnParam	Paraml	Param2	Param3
00001	DEL TSK	E_OK	Recurnraram	H'00000006	Paramz	Paramo
00002	TER_TSK	E_OK		H'00000007	H'0000000	
00003	DEL_TSK	E_OK		H'0000007		
123		10			1	10
				100		
Ready				Read-write	43/65 1 INS	NUM

Figure 2.17 Result of the Manipulation of Objects

(18) Select [Debug->Halt Program] to stop the program execution. To create and initiate MainTask with a priority of 8 that is lower than Task7, select (highlight) task ID 6 in the [Task] window, and then select [Action->Create Task...] from the pop-up menu. This opens the [Create Task] dialog box. Make the entries as shown in figure 2.18.

Attribute C TA_ASM  C TA_HLNG I TA_ACT	Attribute C TA_ASM
C TA <u>A</u> SM C TA_HLNG I⊽ TA_ACT	C TA_ASM  © TA_HLNG IF TA_AQT TA_COP  □  □  □  □  □  □  □  □  □  □ □  4  □  □  □  □  □  □  □  □

Figure 2.18 [Create Task] Dialog Box

(19) Selecting [Hide] from the pop-up menu of the [Task] window or [Action Result] window will close this window. Before doing so, a breakpoint has to be set so that execution of the program will stop at the while(1); statement of Task7. To set a breakpoint, double-click the part circled in figure 2.19.

7729_mix - High-performance Embedded Workshop - [task.c]		- 🗆 🛛
<u>Eile Edit View Project Options Build Debug Memory Tools Window</u>	<u>H</u> elp	- 8 ×
0 15 🔽 💊 47 6 🖪 12 8 2 🛒 🖷 14 🔤 🛛 D 🛎 🖬	0 5 % P 6 {} T 0 <b>]</b> %	<b>A</b> A
🖸 🕰 🚵 🏙 🛗 👗 🔤 obi_big 💽 💽 💽 obi_big_Session		6 3 1
ET EL ET EL ?) (? ( I I I I I I I I I I I I I I I I I	F Tim Int Trp Cyc Alm sVc Rst	
Image: Constraint of the set of	flg(6, 0xffffffff);	• •
XI Stop Stop Build Debug / Find in Files / Version Control		
X Objects	0 10 20 30 40 50	60 🔨
NNL_IDLE KEPNEL OTHERS TSK D'0006 TSK D'0007		
*100Cycle		~
No. Task ID PC Event	Time Stamp	
00001 No Task H'80004450 00002 No Task H'80004450 E_OK 00003 No Task H'80004450 00004 No Task H'80004450 00006 No Task H'80004450 00006 No Task H'80004450 00007 No Task H'80004450 E_OK	0003378747[Cycle] 0003379078[Cycle] 0006706683[Cycle] 0006707768[Cycle] 000670778[Cycle] 0010034620[Cycle] 0010034620[Cycle]	
iiii Ready	Read-write 64/65 1 INS	NUM

Figure 2.19 Setting a Breakpoint

(20) When the execution of the target is resumed, MainTask is initiated, and execution stops at the breakpoint which was set at the while(1); statement of Task7. Check the execution by updating the information displayed for the service-call trace. Since the execution is switched to Task7 after the cre\_tsk service call by MainTask, MainTask has not yet issued the wai\_flg service call.

🖗 7729 mix - High-performance Embedded Workshop - Itask c]	
WP 7729_mbt = Filgr=performance Emicedual Abriency = [Maske] Selie Edit View Project Options Build Debug Memory Tools Window	والله الله الله الله الله الله الله الله
🛛 🛱 🖆 ڭ 🖄 🛗 🕌 👗 🛛 obi_big 📃 💽 obi_big_Session	<u>∧ @</u> ⊑ ⊨ <u>@</u> <u>@</u> ÷ ÷ ± @ <u> </u> ⋈ ₩ # # # #
ET EL ET EL ? ? ? (? @ 4c Tsk Sem FIG Dtg Mbx mTx mBf mPL mi	PF Tim Int Trp Cyc Am sVc Rstt
Image: Specific state         0x8000d024         ercd = set_           Image: Specific state         0x8000d034         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state           Image: Specific state         Image: Specific state         Image: Specific state	flg(6, 0xffffffff);
X Stop PC Breakpoint Build Debug (Find in Files ) Version Control /	
X Objects	0 10 20 30 40 50 60
KNL_IDLE KERNEL OTHERS TSK D'0006 ISK D'0007	
TSK D-0007	
100Cycle	2
No. Task ID PC Event	Time Stamp
00001 No Task H'8000450 cre tsk(0x0006,0x8c009c7c)           00002 No Task H'8000450 E_OK           00003 No Task H'8000450 Task start(0x0000000)           00004 D'0006 H'8000cf98 Task start(0x0000,0x8c021f7c)           00005 D'0006 H'8000cf98 Cre fig(0x0006,0x8c021f7c)           00005 D'0006 H'8000cf98 Task start(0x00000000)           00005 D'0006 H'8000cf98 Cre tsk(0x0007,0x8c021f7c)           00005 D'0006 H'8000cf98 Task start(0x00000000)           00005 D'0007 H'8000d002 Task start(0x000000000)           00008 D'0007 H'8000d002 F8 Task start(0x00000000)           00009 D'0007 H'8000d002 F8 Task start(0x00000000)           00009 D'0007 H'8000d002 F8 Task start(0x00000000)           00009 D'0007 H'8000d002 F8 Task start(0x00000000)	0010036431[Cycle] 0010035437[Cycle] 0010035205[Cycle] 0010036527[Cycle] 0010036527[Cycle] 0010036728[Cycle] 0010037728[Cycle] 0010037925[Cycle] 0010037995[Cycle]
	0020000237(0)2201
Ready	Read-write 63/65 1 INS NUM

Figure 2.20 [RTOS Trace Diagram] Window and [RTOS Trace Text] Window

Rev. 1.0, 08/03, page 24 of 38

# Section 3 Functions

This section is an outline of the functions of the DX. For details on these functions, refer to the online help file.

#### **3.1** Referencing the States of Objects

Selecting an object (such as a task) from the [View->RTOS] menu will display a window which shows the state of each object type, as shown in figure 3.1.

Tsk_ID	Symbol	ExtInf	Priority	State	Wait Factor	Attribute	
0001	tpl	H'00000001	D'0002	WAIT	SLP		
0002	_tp2	H'00000002	D'0002	WAIT	SEM	TA_COP1 TA_COP2	
0003	_tp3	H'00000003	D'0002	WAIT	FLG	- 1999	
0004	_main	H'00000004	D'0001	WAIT	FLG	TA_COP1 TA_COP2	
0005				NOEXS			
0006				NOEXS			
0007	_task_tex_chk	H'00000007	D'0001	WAIT	SLP		
0008	_task_tex_chk	H'00000008	D'0001	WAIT	SLP	TA_COP1 TA_COP2	
0009				NOEXS			
0010	tpl 1	H'0000000A	D'0073	READY		TA COP1 TA COP2	

Figure 3.1 Example Display of Task State

### **3.2** Manipulating the Objects

From the window for each object type, the user can request the manipulation of the object to the kernel. To use this function, the RTOS you are using needs to be configured so as to allow the DX to manipulate the objects.

The manipulation of each object can be requested through the dialog box that can be opened from the pop-up menu of each object window, as shown in figure 3.2.

Tsk_ID:	D' 6	OK
Extended Info.:	[H'0	Cancel
TA_ACT	© TA_HLNG 0 □ 1 □ 2 □ 2 4 □ 5 □ 0 □ 7	
Start A <u>d</u> dress: Initial <u>P</u> riority:	MainTask D' 8 🗨	

Figure 3.2 Example of a Request for Manipulation of an Object

The request for manipulation of the object is sent to the target, and the corresponding service call is issued. The DX will display the results in the [Action Result] window, as shown in figure 3.3.

No.	SVC	Ercd	ReturnParam	Paraml	Param2	Param3	Param4
00001	DEL TSK	E OK		H'00000006			
00002	TER_TSK	E_OK		H'00000007	H'00000000		
00003	DEL_TSK	E_OK		H'00000007			
00004	CRE_TSK	E_OK		H'00000006	H'8C009D54		
00005	VSCR_TSK	E_OK		H'00000001	H'8C009C7C		
00006	STA TSK	E OK		H'00000001	H'00000000		

Figure 3.3 Result of Manipulation of an Object

The processing of the request for manipulation of an object starts when the target is executed. A request can be made for the manipulation of the object after the target has been stopped, but execution will not start until the next execution of the target.

The request for manipulation of the object is queued in the target memory, and the release of the request from the queue is indicated by the display of the result in the [Action Result] window. Up to four requests can be queued.

If manipulation of an object is requested after the target has been stopped, the result is not displayed in the [Action Result] window until the next execution of the target. Accordingly, up to four requests for manipulation of an object can be made at the same time while the target is stopped.

The display in the [Action Result] window is updated with the following timing:

- During execution of the target (cyclic)
- [Refresh] is selected from the pop-up menu displayed by clicking the right-hand mouse button in the [Action Result] window
- User system break (stopped)

## 3.3 Displaying Service Call Trace

When [View->RTOS->Trace] is selected, the history of service calls is displayed in diagrammatic and textual forms, as shown in figure 3.4. To use this function, the RTOS you are using needs to be configured so as to enable display of service call tracing by the DX.

The location of the data that forms the history of service call tracing can be selected as the tool (simulator/debugger or emulator) or the target memory. Depending on the configurator in use, 'Emulator' may be displayed instead of 'Tool'. Note that this configurator can also be used with the simulator/debugger.

🧼 RTOS Trace Diagram 0 10 20 30 40 50 60 70 Objects KNL IDLE KERNEL OTHERS TSK D'0006 TSK D'0007 FLG D'0006 \*100Cvcle RTOS Trace Text 100 No. Task ID PC Event Time Stamp 00001 No Task H'80002f66 cre cyc(0x000b,0x8c000c54) 0000015827[Cycle] 00002 No Task H'80002f66 E OK 0000016262[Cycle] 00003 D'0006 H'8000cf98 Task start (0x0000000) 0000016777[Cycle] 00004 D'0006 H'8000cfb6 cre flg(0x0006,0x8c02217c) 0000017050[Cycle] 00005 D'0006 H'8000cfb6 E\_OK 0000017371[Cycle] 00006 D'0006 H'8000cfe6 cre\_tsk(0x0007,0x8c02217c) 0000017665[Cycle] 00007 D'0006 H'8000cfe6 E OK 0000018530[Cycle] 00009 D'0007 H'8000d020 Task start(0x00000000) 0000019399[Cycle] 00010 D'0007 H'8000d032 set flg(0x0006,0xffffffff) 0000019663[Cycle] 0000020174[Cycle] 00011 D'0006 H'80004000 E OK 00012 D'0006 H'8000d00e del\_flg(0x0006) 0000020440[Cycle] 00013 D'0006 H'8000d00e E OK 0000020757[Cycle] 00014 D'0006 H'8000d018 ext\_tsk() 0000021019[Cycle]

The acquired information can be saved in a file, and this file can be read and displayed.

Figure 3.4 [RTOS Trace Diagram] Window and [RTOS Trace Text] Window

### **3.4** List of Functions

#### 3.4.1 Menu Items

After the DX has been installed, the items listed in table 3.1, which allow accesses to the DX functions, are added to the [View] menu of the HEW.

Pull-Down Menu	Submenu	Function
RTOS	Task	Opens the [Task] window
	Semaphore	Opens the [Semaphore] window
	EventFlag	Opens the [EventFlag] window
	DataQueue	Opens the [DataQueue] window
	Mailbox	Opens the [Mailbox] window
	Mutex	Opens the [Mutex] window
	MessageBuffer	Opens the [MessageBuffer] window
	V-MemoryPool	Opens the [V-MemoryPool] window
	F-MemoryPool	Opens the [F-MemoryPool] window
	Timer	Opens the [Timer] window
	Interrupt	Opens the [Interrupt] window
	Тгар	Opens the [Trap] window
	Cyclic	Opens the [Cyclic] window
	Alarm	Opens the [Alarm] window
	Extended SVC	Opens the [Extended SVC] window
	Action Result	Opens the [Action Result] window
	Trace	Opens the [Trace] window

Table 3.1	Items to be Added to the HEW's [View] Menu
-----------	--

Selecting [View->CPU->Status] displays the DX state in the [Status] window common in HEW.

Note that some functions are not supported depending on the RTOS in use.

#### 3.4.2 Windows and Dialog Boxes

Table 3.2 is a list of windows and dialog boxes for the DX. For further details on the windows and dialog boxes, refer to the online help file. The online help file can be launched by selecting [Help ->RTOS Help].

Classification	Window Name	Functions
Tasks	[Task] window	Displays the states of all tasks
		Displays the ready queue
		Displays locked mutexes for all tasks
	[Task Detail Information] window	Displays detailed information on tasks
	[Create Task] dialog box*	Creates tasks
	[Delete Task] dialog box*	Deletes tasks
	[Activate Task] dialog box*	Initiates tasks
	[Cancel Activate Task] dialog box*	Cancels request for task initiation
	[Start Task] dialog box*	Initiates task (with initiation code specified)
	[Terminate Task] dialog box*	Forcibly terminates tasks
	[Release Wait] dialog box*	Forces tasks out of the wait state
	[Suspend Task] dialog box*	Moves tasks into SUSPENDED state
	[Resume Task] dialog box*	Resumes the execution of tasks in the SUSPENDED state
	[Wakeup Task] dialog box*	Wakes tasks up
	[Cancel Wakeup Task] dialog box*	Cancels wakeup requests
	[Change Task Priority] dialog box*	Changes task priority levels
	[Set Task EventFlag] dialog box*	Sets task-dependent event flags
	[Clear Task EventFlag] dialog box*	Clears task-dependent event flags
	[Rotate Ready Queue] dialog box*	Rotates the ready queue
	[Raise Task Exception Routine] dialog box*	Requests task exception processing
	[Start Overrun Handler] dialog box*	Starts operation of the overrun handler
	[Stop Overrun Handler] dialog box*	Stops the overrun handler

#### Table 3.2Windows and Dialog Boxes

Note: Cannot be used when the RTOS in use was configured without enabling the manipulation of objects.

Classification	Window Name	Functions
Semaphores	[Semaphore] window	Displays states of all semaphores
		Displays the wait task queue for each
		semaphore
	[Increment Semaphore Count] dialog box*	Increments semaphore count
	[Decrement Semaphore Count] dialog box*	Decrements semaphore count
Event flags	[EventFlag] window	Displays the states for all event flags
		<ul> <li>Displays the wait task queue for each event flag</li> </ul>
	[Set EventFlag] dialog box*	Sets an event flag
	[Clear EventFlag] dialog box*	Clears an event flag
Data queue	[DataQueue] window	Displays the states of all data queues
		<ul> <li>Displays the receive data acquired in all data queues</li> </ul>
	[Send Data to DataQueue] dialog box*	Sends data to data queue
	[DataQueue Wait Task Queue] window	Displays the wait task queue for each data queue
Mailboxes	[Mailbox] window	Displays the states of all mailboxes
		<ul> <li>Displays the wait task queue for each mailbox</li> </ul>
		Displays the received message
		queues for all mailboxes
	[Send Message to Mailbox] dialog box*	Sends messages to mailboxes
	[Receive Message from Mailbox] dialog box*	Receive messages from mailboxes
Mutex	[Mutex] window	Displays the states of all mutexes
		<ul> <li>Displays the wait task queue for each mutex</li> </ul>

### Table 3.2 Windows and Dialog Boxes (cont)

Note: Cannot be used when the RTOS in use was configured without enabling the manipulation of objects.

Classification	Window Name	Functions
Message buffer	[MessageBuffer] window	Displays the states of all message buffers
		Displays the wait task queue for each message buffer
		Displays the received message queue for each message buffer
	[Send Message to MessageBuffer] dialog box*	Sends messages to message buffers
Variable-size memory pool	[Variable-size MemoryPool] window	Displays the states of all variable- size memory pools
		Displays the wait task queue for each variable-size memory pool
	[Get Variable-size MemoryPool] dialog box*	Acquires variable-size memory blocks
	[Release Variable-size MemoryPool] dialog box*	Releases variable-size memory blocks
Fixed-size memory pool	[Fixed-size MemoryPool] window	Displays the states of all fixed-size memory pools
		Displays the wait task queue for each fixed-size memory pool
	[Get Fixed-size MemoryPool] dialog box*	Acquires fixed-size memory blocks
	[Release Fixed-size MemoryPool] dialog box*	Releases fixed-size memory blocks
Timer	[Timer] window	Displays the system clock setting
		Displays the wait queue for the timer
	[Set Time] dialog box*	Modifies the system clock setting
Interrupt handler, exception processing routine	[Interrupt Handler] window	Displays the states of all interrupt handlers and exception processing routines
Trap exception processing routine	[Trap Routine] window	Displays the states of all trap exception processing routines

### Table 3.2 Windows and Dialog Boxes (cont)

Rev. 1.0, 08/03, page 32 of 38

objects.

Classification	Window Name	Functions
Cyclic handler	[Cyclic Handler] window	Displays the states of all cyclic handlers
	[Start Cyclic Handler] dialog box*	Starts up cyclic handlers
	[Stop Cyclic Handler] dialog box*	Stops cyclic handlers
Alarm handler	[Alarm Handler] window	Displays the state of all alarm handlers
	[Start Alarm Handler] dialog box*	Starts up alarm handlers
	[Stop Alarm Handler] dialog box*	Stops alarm handlers
Extended SVC routine	[Extended SVC Routine] window	Displays the states of all extended SVC routines
Result	[Action Result] window*	Displays the results of object manipulation
Trace	[RTOS Trace Diagram] window	Displays the object item list
		<ul> <li>Displays trace information (diagrammatic form)</li> </ul>
	[RTOS Trace Text] window	Displays the object item list
		<ul> <li>Displays trace information (textual form)</li> </ul>
	[RTOS Trace Options] dialog box	Makes various settings for trace
	[Select Diagram Object] dialog box	Defines the target object for trace display in the diagrammatic form
	[Select Text Object] dialog box	Defines trace information for the kernel resource
	[Load RTOS Trace Information File] dialog box	Inputs the trace information file
	[RTOS Trace Statistics] dialog box	Displays overall view of program execution
	[RTOS Trace Find] window	Searches for event information
	[Save As] dialog box	Outputs trace information to a file

### Table 3.2 Windows and Dialog Boxes (cont)

Note: Cannot be used when the RTOS in use was configured without enabling the manipulation of objects.

Rev. 1.0, 08/03, page 34 of 38

# Section 4 Precautions and Restrictions

### 4.1 Realtime Operation of the User System and Cyclic Handler

The DX functions are realized by referencing or updating data in the memory of the user system. If the following functions are used during the execution of the user program, memory access will occur, and operation will not be in realtime.

- Opening or updating a window
- Clicking the [OK] button of a dialog box to manipulate an object

If the RTOS in use was configured with the manipulation of objects enabled, the throughput of the user system falls a little because a cyclic handler for the DX with a certain cycle is automatically installed in the system.

## 4.2 Consistency of Displayed Contents

Reference to an object's state is by directly reading from data in the memory of the user system. Therefore, if such data is read during the execution of an RTOS kernel routine, the information displayed may not be correct. Also, if an object's state is referred to before the initialization of the kernel has been completed, the information displayed may not be correct.

Rev. 1.0, 08/03, page 36 of 38

# Appendix A Troubleshooting

### A.1 Displaying [Action Result] Window Results

If an object manipulation request is made through a dialog box and the user program is executed but the display is not updated, this could be because the cyclic handler for the DX is not operating correctly.

The following is a list of causes and the measures to be taken, in such cases.

- 1. No timer interrupt occurs.
  - A. Timer interrupts do not occur.

Correct the timer driver so that timer interrupts will occur.

B. Timer interrupts do occur but are not accepted.

In the following cases, timer interrupts are not accepted.

- a. Too many interrupts occur that have an interrupt level higher than the timer interrupt
- b. The timer interrupt level has been masked for a long time.

For interrupts that have a level equal to or higher than the timer interrupt level, keep the masking time as short as possible.

2. The system goes down.

When the system goes down, timer interrupts may not be accepted.

Take care so that the system will not go down.

Rev. 1.0, 08/03, page 38 of 38

# **Debugging Extension User's Manual**

Publication Date:	Rev.1.00, August 5, 2003
Published by:	Sales Strategic Planning Div.
	Renesas Technology Corp.
Edited by:	Technical Documentation & Information Department Renesas Kodaira Semiconductor Co., Ltd.

©2003 Renesas Technology Corp. All rights reserved. Printed in Japan.

# Debugging Extension User's Manual



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

REJ10B0029-0100H