E1 On-Chip Debugging Emulator

Introductory Guide for the RX610 Group

R20AN0022EJ0100
Rev.1.00
Sep. 07, 2010

Outline

This introductory guide is intended for first-time users of the E1 emulator. It provides simple instructions to help the user along the path from unpacking the E1 emulator to running and stopping a program without getting lost along the way.

Contents

1. Introduction

This guide assumes that the user who purchased the E1 emulator as a unit is using a CPU board on which an RX610-group MCU has been installed as the user system. However, the same operating procedures for the E1 emulator can also be used with systems which incorporate other RX610-group MCUs.

By following the procedures in this guide, you can get some practical experience of installing the program and simple usage of the E1 emulator.

This guide assumes that the machines and tools in use are as follows.

(1) Host computer
(2) E1 emulator
(3) CPU board on which an RX610-group MCU has been installed (R0K556100C000BE):
This CPU board is included in the Renesas Starter Kit for the RX610 (R0K556100S000BE).

2. Product Overview

Figure 2.1 shows the configuration of the E1 emulator system.

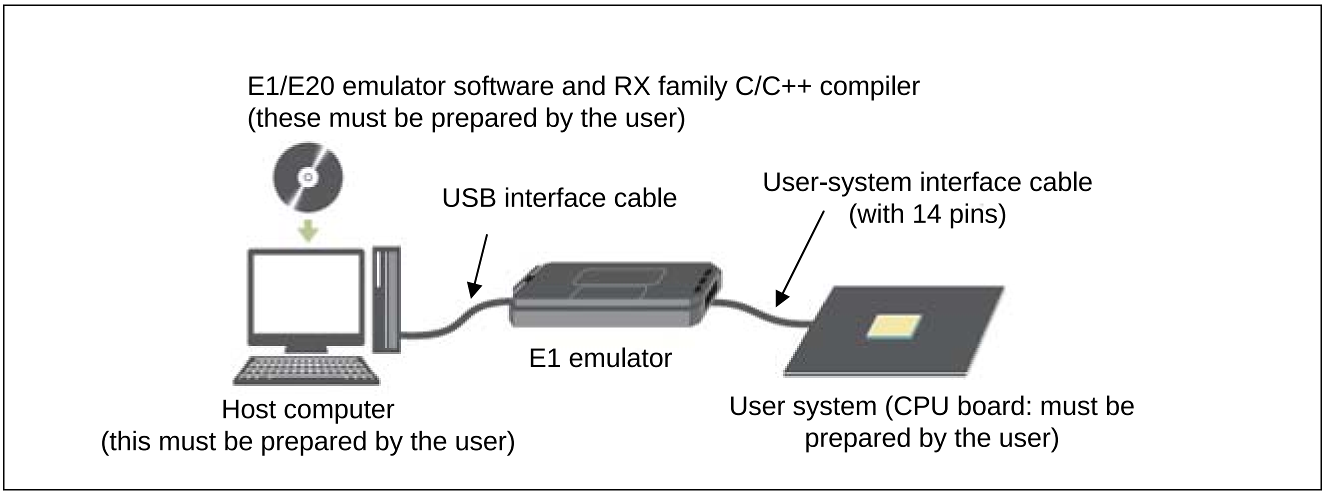

E1/E20 emulator software and RX family C/C++ compiler
(these must be prepared by the user)
USB interface cable
User-system interface cable
(with 14 pins)
E1 emulator
Host computer
(this must be prepared by the user)
User system (CPU board: must be
prepared by the user)

Figure 2.1 System Configuration with the Emulator

2.1 Items Included in the E1 Emulator Package

2.1.1 E1 Emulator

The E1 is an on-chip debugging emulator from Renesas that provides a highly-efficient debugging environment in combination with the High-performance Embedded Workshop and E1/E20 emulator debugger.

After debugging has been completed, the E1 is also usable as a flash-memory programmer in combination with the Flash Development Toolkit (FDT).

2.1.2 USB Interface Cable

This cable is used to connect the emulator and host computer.

2.1.3 User-System Interface Cable

This cable is used to connect the emulator and user system.

2.1.4 E1/E20 Emulator Software

The E1/E20 emulator software consists of the following items.

(1) High-performance Embedded Workshop

The High-performance Embedded Workshop is a common GUI in which various tools such as a C compiler, assembler, emulator software, and editor are integrated for the more efficient development of software. Debugging with the E1 emulator is always controlled via this GUI.

(2) E1/E20 emulator debugger

The E1/E20 emulator debugger runs on the host computer and communicates with the MCU via the E1 emulator, providing an advanced debugging environment. Users are not directly aware of the presence of the emulator debugger because it operates as part of the High-performance Embedded Workshop. The features of the E1 emulator debugger are given below.

- Basic debugging functions including running and stopping a program, breaks in execution, and watching
- Acquiring trace information on 256 branches or cycles
- Measuring performance between two points (i.e. performance measurement facility)
- Automatic updating of the display of data in the [Memory] window by using a cycle-stealing transfer mode

These enhanced debugging functions facilitate the analysis of problems during debugging and evaluation, greatly reducing times required for debugging.

(3) Auto-update utility

The auto-update utility automatically acquires update information on Renesas' microcomputer development tools via the Internet to ensure that required updates are applied to the tools.

For details, see the Renesas AutoUpdate Manual, which is opened from the corresponding menu item shown in Figure 2.2.

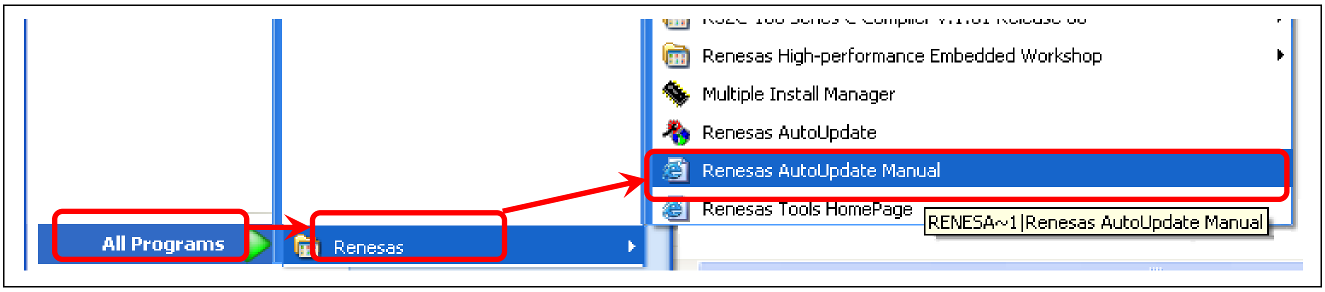

Renesas High-performance Embedded Workshop
Multiple Install Manager
Renesas AutoUpdate
Renesas AutoUpdate Manual
Renesas Tools HomePage
RENESA~1|Renesas AutoUpdate Manual
All Programs
Renesas

Figure 2.2 Opening the Renesas AutoUpdate Manual

2.2 Items That Must be Prepared by the User

The following items must be prepared by the user.

- Host computer
- User system (CPU board)
- RX family C/C++ compiler package*

*Note: There are two versions of the RX family C/C++ compiler package: official product and free evaluation software. For details on the evaluation version of the C compiler, refer to section 5.2, Evaluation Version of the RX Family C/C++.

2.2.1 Host Computer

The versions of OS and host computer we recommend for running the software products included in the E1 emulator package are given in Table 2.1.

Table 2.1 Operating Environments

Item	Operating Environments	
	Windows® 2000 or 32-Bit Editions of Windows® XP	32-Bit Editions of Windows Vista®
Host computer	IBM PC/AT compatible machine with USB 2.0 (Full-Speed/High-Speed*).	
CPU	At least a 1-GHz Pentium® III is recommended.	At least a 3-GHz Pentium® 4 or 1-GHz Core™ 2 Duo is recommended.
Minimum memory capacity	1 Gbyte or more is recommended (at least 10 times the size of the load module file).	1.5 Gbyte or more is recommended (at least 10 times the size of the load module file).
Hard-disk capacity	Installation disk capacity: 600 Mbytes or more. Prepare a swap area taking up at least double the memory capacity (four times or more is recommended).	
Pointing device such as mouse	Connectable to the host computer; compatible with Windows® 2000, Windows® XP, or Windows Vista®.	
Display	Monitor resolution: 1024 × 768 or higher	
CD-ROM drive	Required to install the emulator software or refer to the emulator user's manual.	

*Note: Although the emulators are also connectable to a host computer with USB 1.1, we recommend USB 2.0 (High-Speed) as more suitable in terms of emulator performance.

2.2.2 User System

The E1 emulator must always be used with a user system on which an MCU has been installed. The user system must have a connector for the E1.

In this guide, the CPU board (R0K556100C000BE) included in the Renesas Starter Kit for the RX610 (R0K556100S000BE) is used instead of a user system.

The same procedures as are given in this guide apply to any CPU board on which a RX610-group MCU has been installed.

2.2.3 RX Family C/C++ Compiler Package

The compiler is not included in the E1 emulator package. To compile source files, prepare the RX family C/C++ compiler package.

The RX family C/C++ compiler package is capable of generating debugging information files from C- and assembly-language programs.

3. Installation

The following products must be installed if the E1 emulator is to be used.

- E1/E20 emulator software (install this from the CD included in the E1 emulator package)
  This product includes the E1 emulator debugger, integrated development environment (High-performance Embedded Workshop), and auto-update utility.
- RX family C/C++ compiler package
  This product includes the C/C++ compiler suite, simulator/debugger, integrated development environment (High-performance Embedded Workshop), and auto-update utility.

3.1 Before Installation

(1) Do not connect the E1 emulator to the host computer before the emulator software is installed. Installing the emulator software involves installation of the E1-specific USB driver on the host computer. This automatically provides guidance when new hardware is detected.

If the E1 emulator is connected to the host computer before the emulator software is installed, the host computer will not be able to recognize the E1 emulator because the E1-specific USB driver is not present. In such cases, use the Windows' Device Manager to delete the unidentified USB device set up in response to the connection. After that, install the emulator software and reconnect the E1 emulator.

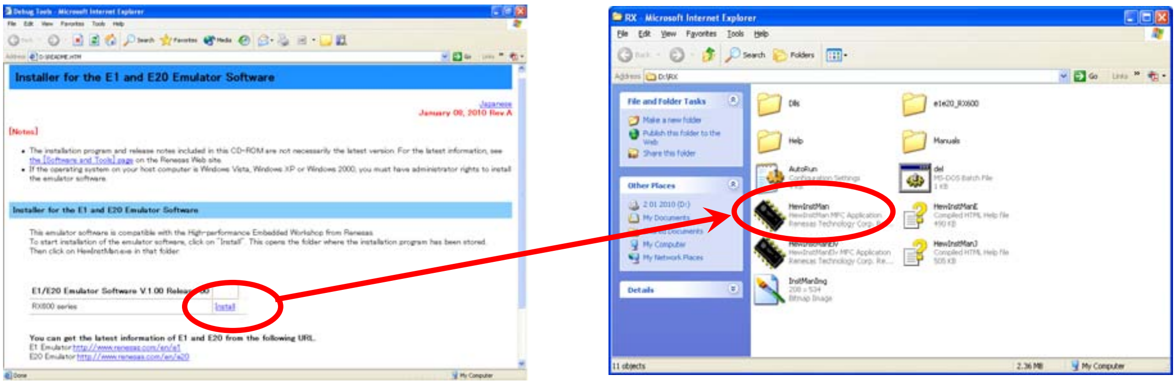
(2) If you have both the official product and evaluation version of the software, use the product version.

Also use the product version if you have installed the product version of the RX-series C compiler package or purchased the compiler package along with the E1 emulator.

3.2 Installing the Emulator Software

(1) Insert the CD-ROM included in the E1 emulator package into the host computer. An HTML file giving instructions will automatically be opened (if this does not happen, open README_E. HTM on the CD-ROM). Click on the [Install] link shown in Figure 3.1.

(2) The contents of a folder are displayed as shown in Figure 3.1. Execute HewInstMan.exe.



Installer for the E1 and E20 Emulator Software

Japanese
January 08, 2010 Rev A

[Notes]

- The installation program and release notes included in this CD-ROM are not necessarily the latest version. For the latest information, see the [Software and Tools] page on the Renesas Web site.
- If the operating system on your host computer is Windows Vista, Windows XP or Windows 2000, you must have administrator rights to install the emulator software.

Installer for the E1 and E20 Emulator Software

This emulator software is compatible with the High-performance Embedded Workshop from Renesas.
To start installation of the emulator software, click on "Install". This opens the folder where the installation program has been stored.
Then click on HewInstMan.exe in that folder.

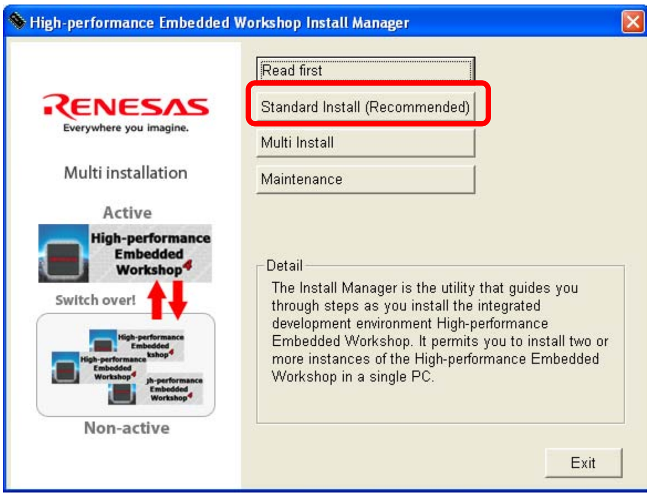
E1/E20 Emulator Software V.1.00 Release 00	
RX600 series	Install

You can get the latest information of E1 and E20 from the following URL.
E1 Emulator http://www.renesas.com/en/e1
E20 Emulator http://www.renesas.com/en/e20

Figure 3.1 Introductory Screen for Installation

(3) Executing HewInstMan.exe opens the [High-performance Embedded Workshop Install Manager] window shown in Figure 3.2.

On the first installation, [Install Manager Help] is also displayed, so confirm its content. After that, click on [Standard Install (Recommended)].

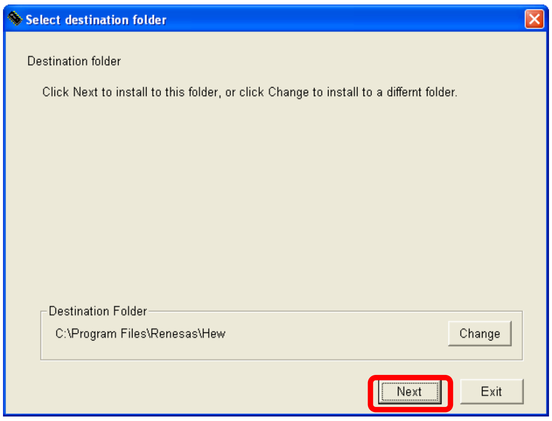


The Install Manager is the utility that guides you through steps as you install the integrated development environment High-performance Embedded Workshop. It permits you to install two or more instances of the High-performance Embedded Workshop in a single PC.

Figure 3.2 High-performance Embedded Workshop Installation Manager

Note: Although this guide explains the standard installation, you can alternatively select [Multi Install] to install multiple High-performance Embedded Workshops on a single host computer.

(4) The [Select destination folder] dialog box appears (Figure 3.3). To change the destination folder, click on the [Change] button. Check the folder name and click on the [Next] button. The default destination folder is used in this guide.



Destination folder

Click Next to install to this folder, or click Change to install to a differnt folder.

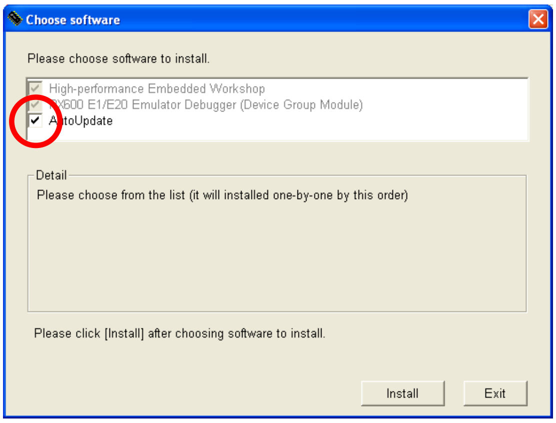
Destination Folder

C:\Program Files\Renesas\Hew

Figure 3.3 [Select destination folder]

(5) The [Choose software] dialog box appears (Figure 3.4). Select the software products that you wish to install and click on the [Install] button.

The auto-update utility, which automatically acquires update information on Renesas' microcomputer development tools via the Internet to ensure that required updates are applied to the tools, is selected in this guide.



Please choose software to install.

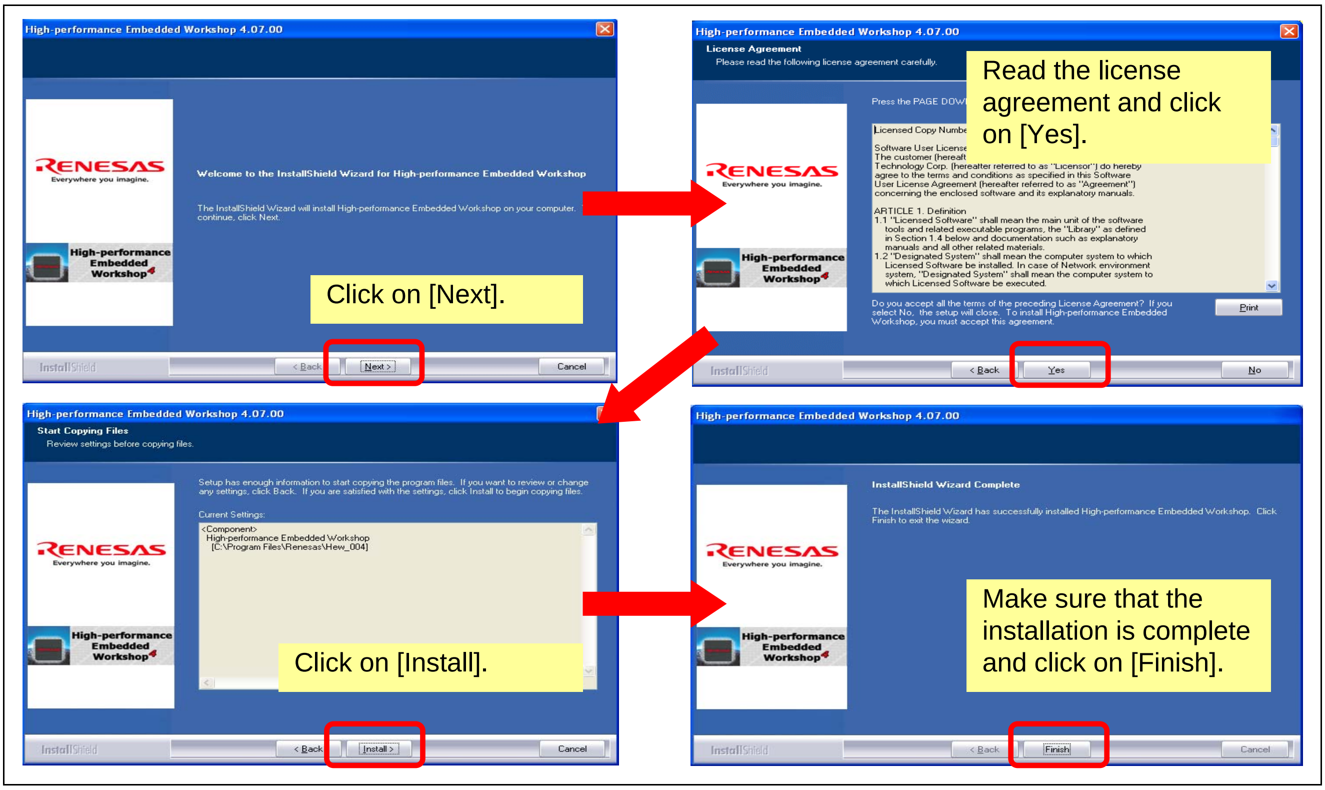
☑ High-performance Embedded Workshop
☑ RX600 E1/E20 Emulator Debugger (Device Group Module)
☑ AutoUpdate

Detail
Please choose from the list (it will installed one-by-one by this order)

Please click [Install] after choosing software to install.

Figure 3.4 [Choose software]

(6) Installation of the High-performance Embedded Workshop starts automatically. Follow the procedures shown in Figure 3.5.



High-performance Embedded Workshop 4.07.00

Welcome to the InstallShield Wizard for High-performance Embedded Workshop

The InstallShield Wizard will install High-performance Embedded Workshop on your computer. To continue, click Next.

Click on [Next].

High-performance Embedded Workshop 4.07.00

License Agreement

Please read the following license agreement carefully.

Press the PAGE DOW

Licensed Copy Numbe

Software User License
The customer (hereaft
Technology Corp. (hereafter referred to as "Licensor") do hereby
agree to the terms and conditions as specified in this Software
User License Agreement (hereafter referred to as "Agreement")
concerning the enclosed software and its explanatory manuals.

ARTICLE 1. Definition
1.1 "Licensed Software" shall mean the main unit of the software
tools and related executable programs, the "Library" as defined
in Section 1.4 below and documentation such as explanatory
manuals and all other related materials.
1.2 "Designated System" shall mean the computer system to which
Licensed Software be installed. In case of Network environment
system, "Designated System" shall mean the computer system to
which Licensed Software be executed.

Do you accept all the terms of the preceding License Agreement? If you select No, the setup will close. To install High-performance Embedded Workshop, you must accept this agreement.

Read the license agreement and click on [Yes].

High-performance Embedded Workshop 4.07.00

Start Copying Files

Review settings before copying files.

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Install to begin copying files.

Current Settings:

```
<Component>
 High-performance Embedded Workshop
  [C:\Program Files\Renesas\Hew_004]
```

Click on [Install].

High-performance Embedded Workshop 4.07.00

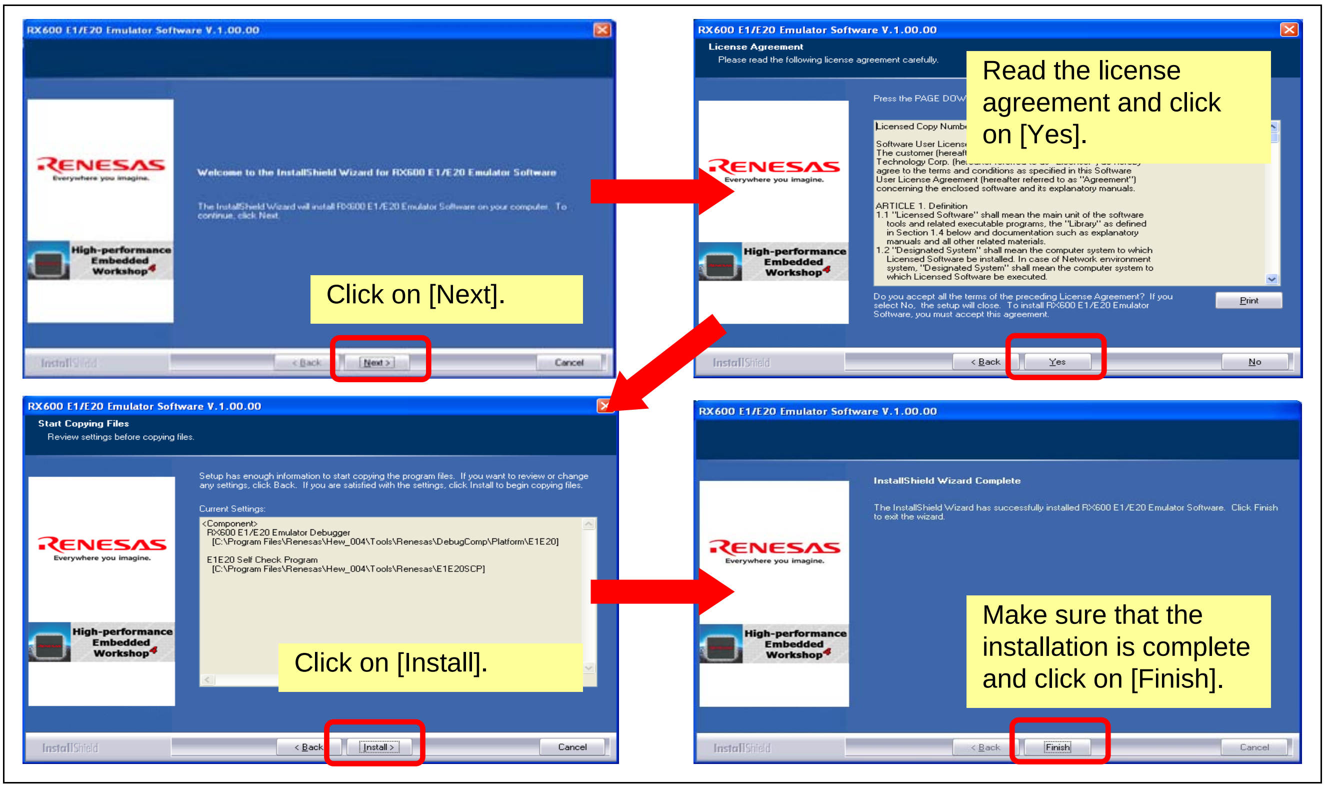
InstallShield Wizard Complete

The InstallShield Wizard has successfully installed High-performance Embedded Workshop. Click Finish to exit the wizard.

Make sure that the installation is complete and click on [Finish].

Figure 3.5 Installing the High-performance Embedded Workshop

(7) Installation of the E1/E20 emulator software starts. Follow the procedure shown in Figure 3.6.



RX600 E1/E20 Emulator Software V.1.00.00

Welcome to the InstallShield Wizard for RX600 E1/E20 Emulator Software

The InstallShield Wizard will install RX600 E1/E20 Emulator Software on your computer. To continue, click Next.

Click on [Next].

RX600 E1/E20 Emulator Software V.1.00.00

License Agreement

Please read the following license agreement carefully.

Press the PAGE DOW

Licensed Copy Numbe

Software User License
The customer (hereaft
Technology Corp. (he
agree to the terms and conditions as specified in this Software
User License Agreement (hereafter referred to as "Agreement")
concerning the enclosed software and its explanatory manuals.

ARTICLE 1. Definition
1.1 "Licensed Software" shall mean the main unit of the software
tools and related executable programs, the "Library" as defined
in Section 1.4 below and documentation such as explanatory
manuals and all other related materials.
1.2 "Designated System" shall mean the computer system to which
Licensed Software be installed. In case of Network environment
system, "Designated System" shall mean the computer system to
which Licensed Software be executed.

Do you accept all the terms of the preceding License Agreement? If you select No, the setup will close. To install RX600 E1/E20 Emulator Software, you must accept this agreement.

Read the license agreement and click on [Yes].

RX600 E1/E20 Emulator Software V.1.00.00

Start Copying Files

Review settings before copying files.

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Install to begin copying files.

Current Settings:

```
<Component>
 RX600 E1/E20 Emulator Debugger
  [C:\Program Files\Renesas\Hew_004\Tools\Renesas\DebugComp\Platform\E1E20]

 E1E20 Self Check Program
  [C:\Program Files\Renesas\Hew_004\Tools\Renesas\E1E20SCP]
```

Click on [Install].

RX600 E1/E20 Emulator Software V.1.00.00

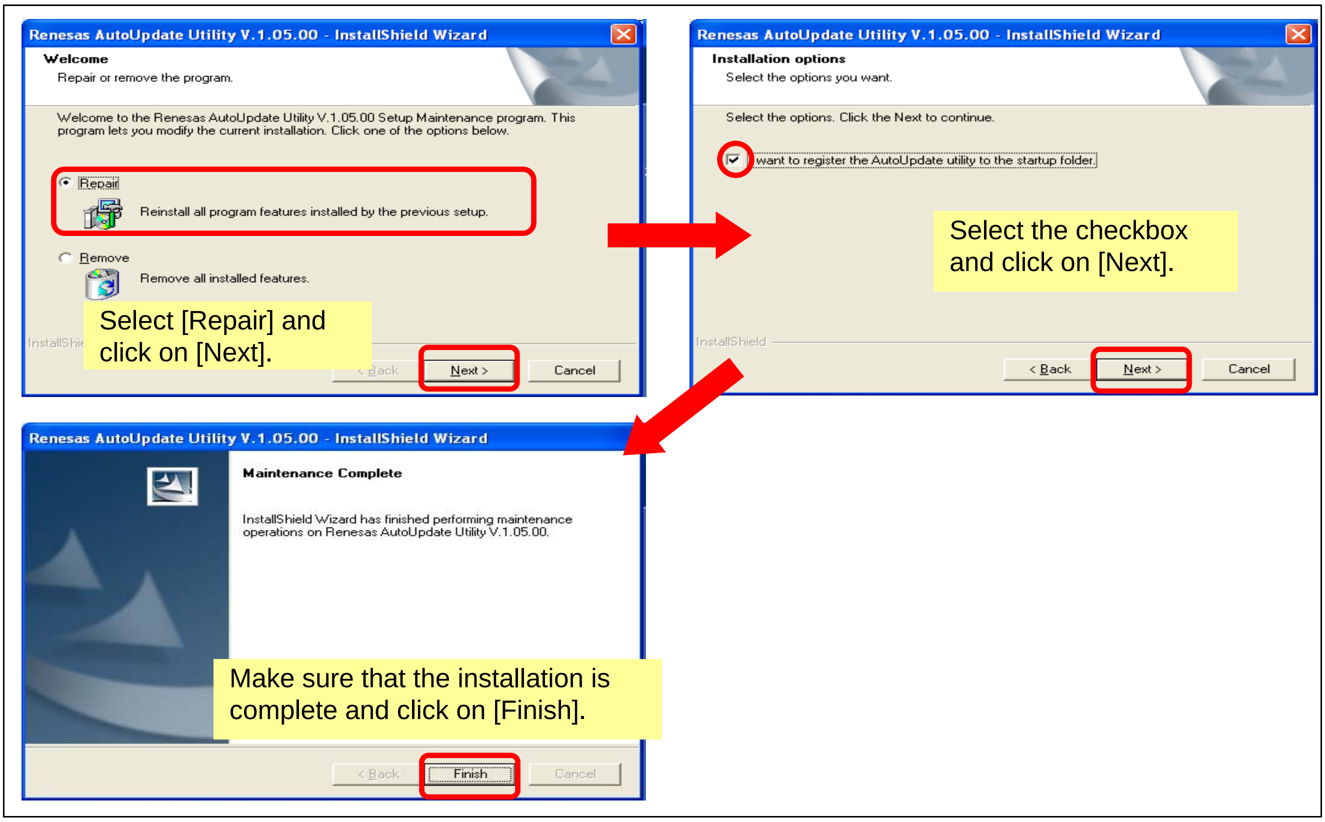
InstallShield Wizard Complete

The InstallShield Wizard has successfully installed RX600 E1/E20 Emulator Software. Click Finish to exit the wizard.

Make sure that the installation is complete and click on [Finish].

Figure 3.6 Installing the E1/E20 Emulator Software

(8) Installation of the auto-update utility starts. Follow the procedure shown in Figure 3.7.



Renesas AutoUpdate Utility V.1.05.00 - InstallShield Wizard

Welcome

Repair or remove the program.

Welcome to the Renesas AutoUpdate Utility V.1.05.00 Setup Maintenance program. This program lets you modify the current installation. Click one of the options below.

● Repair

Reinstall all program features installed by the previous setup.

○ Remove

Remove all installed features.

Select [Repair] and click on [Next].

Renesas AutoUpdate Utility V.1.05.00 - InstallShield Wizard

Installation options

Select the options you want.

Select the options. Click the Next to continue.

☑ I want to register the AutoUpdate utility to the startup folder.

Select the checkbox and click on [Next].

Renesas AutoUpdate Utility V.1.05.00 - InstallShield Wizard

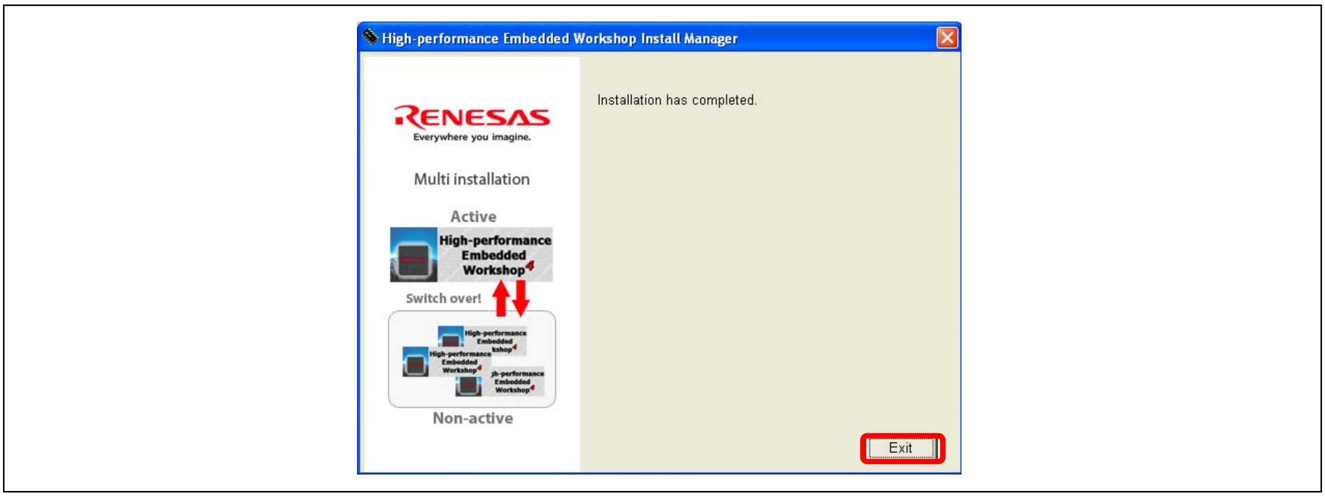
Maintenance Complete

InstallShield Wizard has finished performing maintenance operations on Renesas AutoUpdate Utility V.1.05.00.

Make sure that the installation is complete and click on [Finish].

Figure 3.7 Installing the Auto-Update Utility

(9) The dialog box shown in Figure 3.8 appears when the installation of all software products has been completed. Click on the [Exit] button to finish this process.



High-performance Embedded Workshop Install Manager

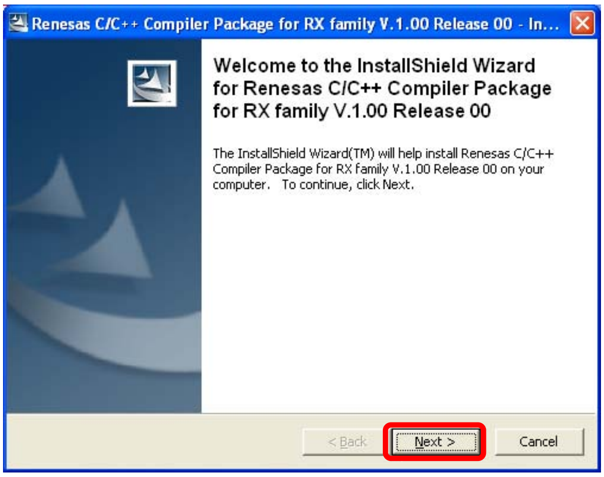
Installation has completed.

Figure 3.8 Installation Completed

3.3 Installing the RX Family C/C++ Compiler Package

(1) To install the evaluation version of the compiler, download the executable file (e.g. ccrxv100r00_ev.exe) and double-click on the filename or icon. In the dialog box shown in Figure 3.9, click on the [Next] button. To install an official product-version compiler, insert the CD into the computer drive. The installation process starts with the screen shown in Figure 3.10.

For details on downloading of the evaluation version of the compiler, refer to section 5.2.2, Downloading the Evaluation-Version C Compiler.



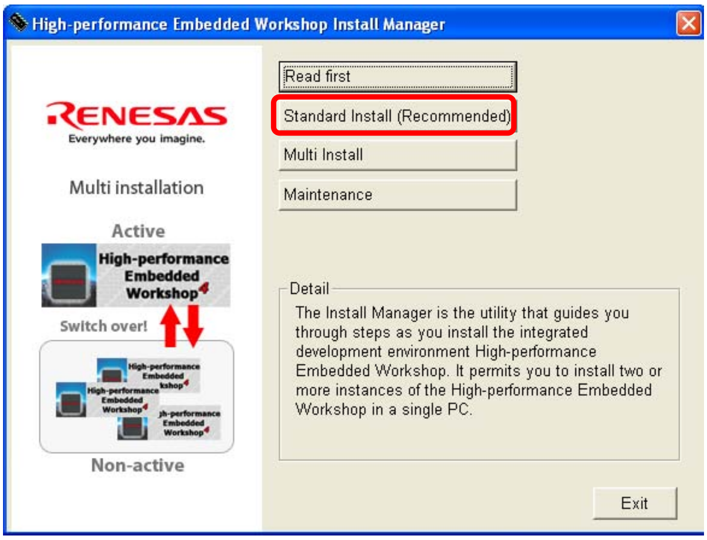
Renesas C/C++ Compiler Package for RX family V.1.00 Release 00 - In...

Welcome to the InstallShield Wizard for Renesas C/C++ Compiler Package for RX family V.1.00 Release 00

The InstallShield Wizard(TM) will help install Renesas C/C++ Compiler Package for RX family V.1.00 Release 00 on your computer. To continue, click Next.

Figure 3.9 Starting to Install the Evaluation-Version C Compiler

(2) The dialog box shown in Figure 3.10 appears. Select [Standard Install (Recommended)].



High-performance Embedded Workshop Install Manager

Detail

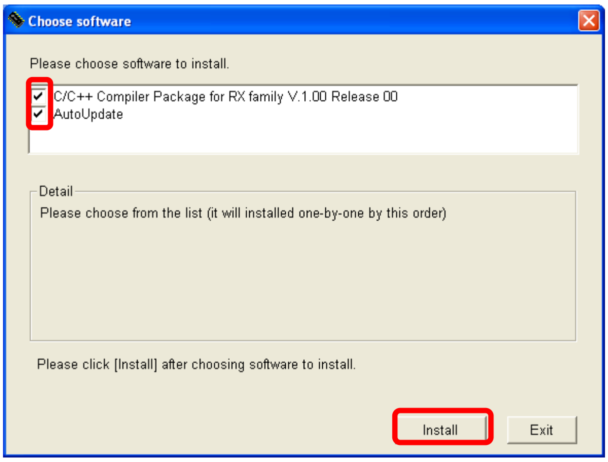
The Install Manager is the utility that guides you through steps as you install the integrated development environment High-performance Embedded Workshop. It permits you to install two or more instances of the High-performance Embedded Workshop in a single PC.

Figure 3.10 High-performance Embedded Workshop Installation Manager

(3) Select the software products that you wish to install and click on the [Install] button.

At least [C/C++ Compiler Package for RX family] must be selected.

Installation of the auto-update utility is optional. Although there is no need to select the auto-update utility if it has already been installed on the host computer, selecting it causes no problems since the newer version of the auto-update utility will then be on the host computer.



Please choose software to install.

☑ C/C++ Compiler Package for RX family V.1.00 Release 00
☑ AutoUpdate

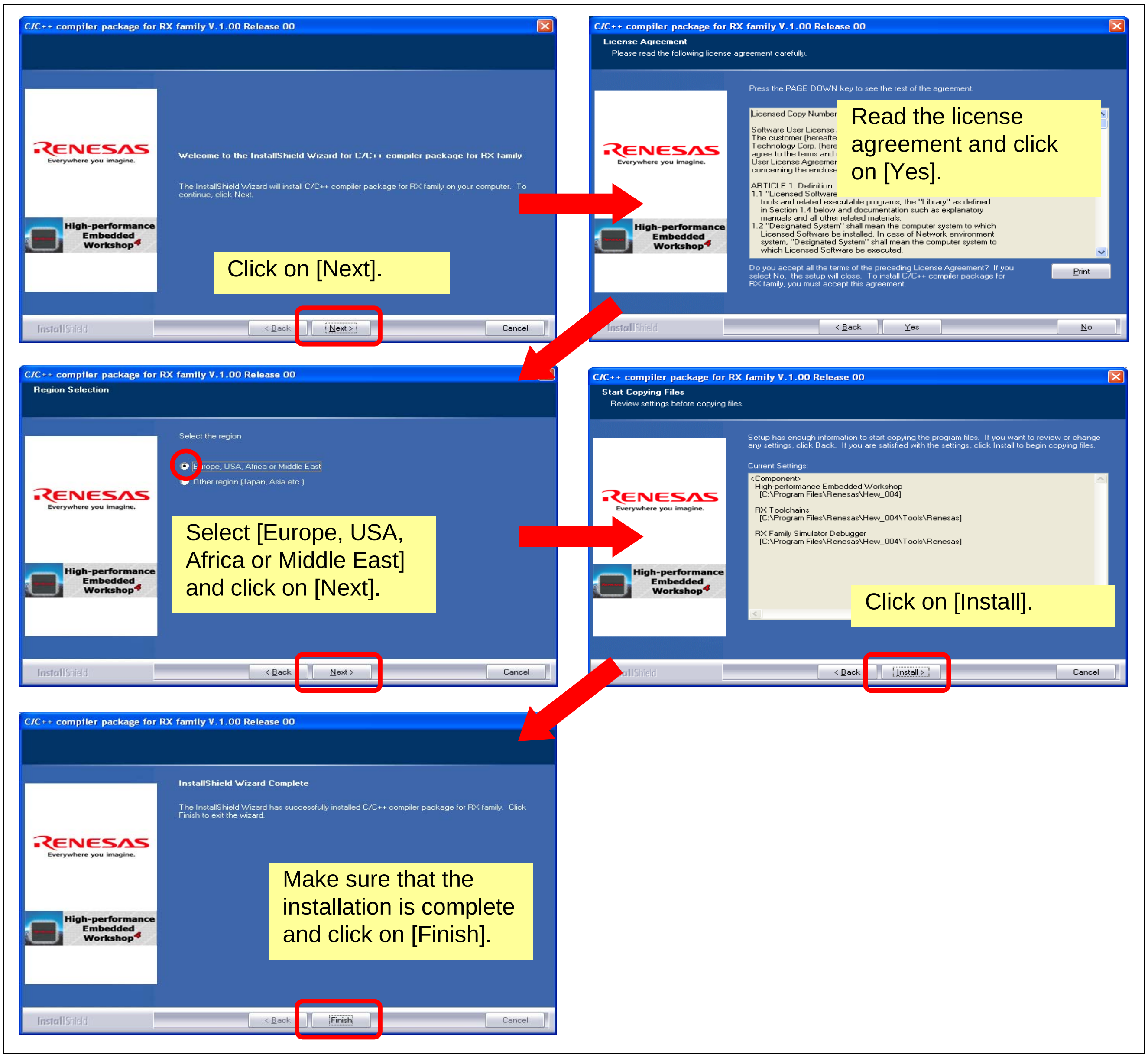
Detail

Please choose from the list (it will installed one-by-one by this order)

Please click [Install] after choosing software to install.

Figure 3.11 Selecting Software Products to be Installed

(4) Installation of the C/C++ compiler package starts. Follow the procedure shown in Figure 3.12.



C/C++ compiler package for RX family V.1.00 Release 00

Welcome to the InstallShield Wizard for C/C++ compiler package for RX family

The InstallShield Wizard will install C/C++ compiler package for RX family on your computer. To continue, click Next.

Click on [Next].

License Agreement

Please read the following license agreement carefully.

Press the PAGE DOWN key to see the rest of the agreement.

Licensed Copy Number

Software User License
The customer (hereafte
Technology Corp. (here
agree to the terms and
User License Agreemer
concerning the enclose

ARTICLE 1. Definition
1.1 "Licensed Software
tools and related executable programs, the "Library" as defined in Section 1.4 below and documentation such as explanatory manuals and all other related materials.
1.2 "Designated System" shall mean the computer system to which Licensed Software be installed. In case of Network environment system, "Designated System" shall mean the computer system to which Licensed Software be executed.

Do you accept all the terms of the preceding License Agreement? If you select No, the setup will close. To install C/C++ compiler package for RX family, you must accept this agreement.

Read the license agreement and click on [Yes].

Region Selection

Select the region

● Europe, USA, Africa or Middle East
○ Other region (Japan, Asia etc.)

Select [Europe, USA, Africa or Middle East] and click on [Next].

Start Copying Files

Review settings before copying files.

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Install to begin copying files.

Current Settings:

```
<Component>
 High-performance Embedded Workshop
  [C:\Program Files\Renesas\Hew_004]

 RX Toolchains
  [C:\Program Files\Renesas\Hew_004\Tools\Renesas]

 RX Family Simulator Debugger
  [C:\Program Files\Renesas\Hew_004\Tools\Renesas]
```

Click on [Install].

InstallShield Wizard Complete

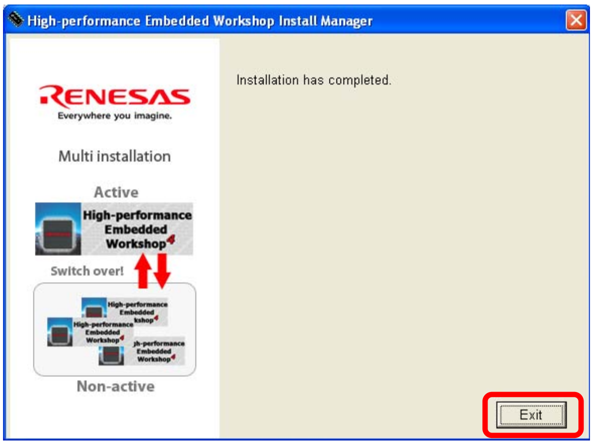
The InstallShield Wizard has successfully installed C/C++ compiler package for RX family. Click Finish to exit the wizard.

Make sure that the installation is complete and click on [Finish].

Figure 3.12 Installing the C/C++ Compiler Package

(5) The auto-update utility is also automatically installed when it has been selected in the [Choose software] dialog box. Follow the instructions shown on the screen. Details on the installation process are omitted here.

(6) The dialog box shown in Figure 3.13 appears when the installation of the compiler package has been completed. Click on the [Exit] button to finish this process.

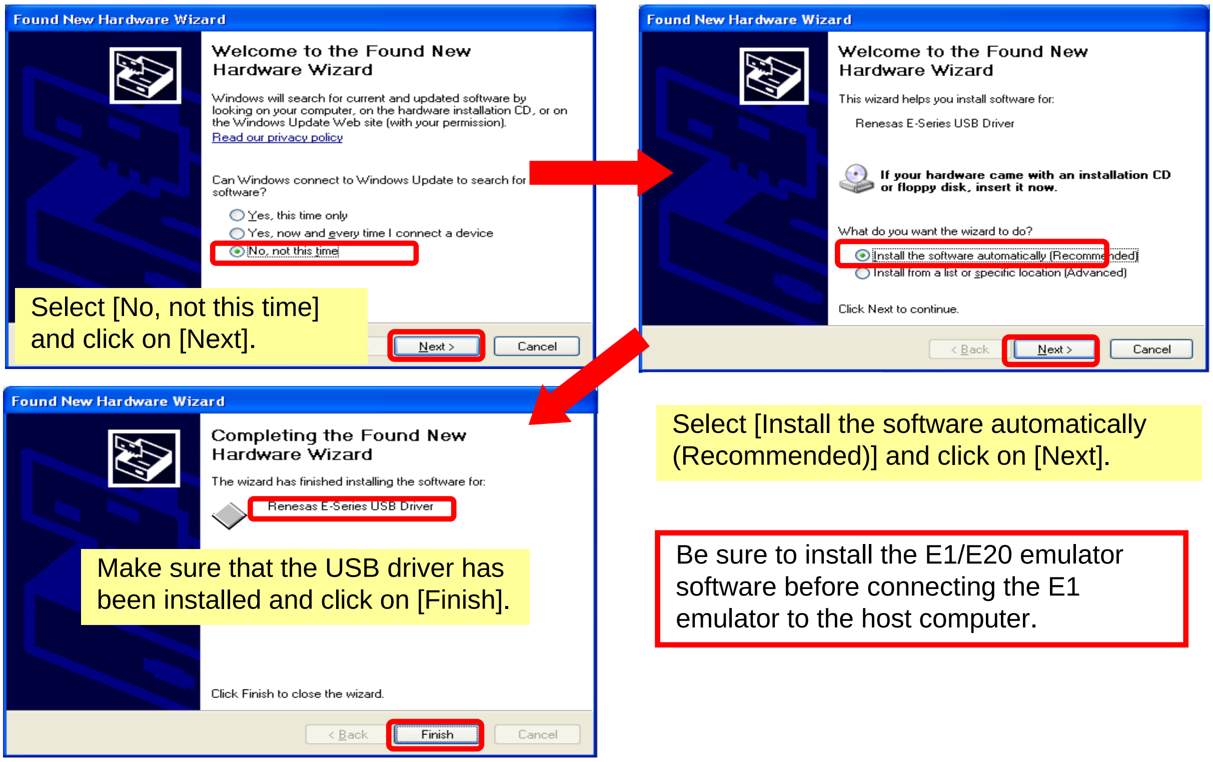

High-performance Embedded Workshop Install Manager
RENESAS
Everywhere you imagine.
Multi installation
Active
High-performance Embedded Workshop
Switch over!
Non-active
Exit

Installation has completed.

Figure 3.13 Installation Completed

3.4 Installing the USB Driver

(1) Connect the E1 emulator and host computer via a USB interface cable. When the dialog box shown in Figure 3.14 appears, follow the instructions on the screen to install the USB driver.
The pictures below are from a computer running Windows® XP. The display varies with the operating system.



Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

Read our privacy policy

Can Windows connect to Windows Update to search for software?

○ Yes, this time only
○ Yes, now and every time I connect a device
● No, not this time

Select [No, not this time] and click on [Next].

Welcome to the Found New Hardware Wizard

This wizard helps you install software for:

Renesas E-Series USB Driver

If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

● Install the software automatically (Recommended)
○ Install from a list or specific location (Advanced)

Click Next to continue.

Select [Install the software automatically (Recommended)] and click on [Next].

Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

Renesas E-Series USB Driver

Click Finish to close the wizard.

Make sure that the USB driver has been installed and click on [Finish].

Be sure to install the E1/E20 emulator software before connecting the E1 emulator to the host computer.

Figure 3.14 Found New Hardware Wizard

3.5 Checking That the Emulator Debugger and Compiler Have been Installed

Follow the procedure below to check that the emulator debugger and compiler have successfully been installed.

(1) Select [Multiple Install Manager] from the start menu as shown in Figure 3.15.

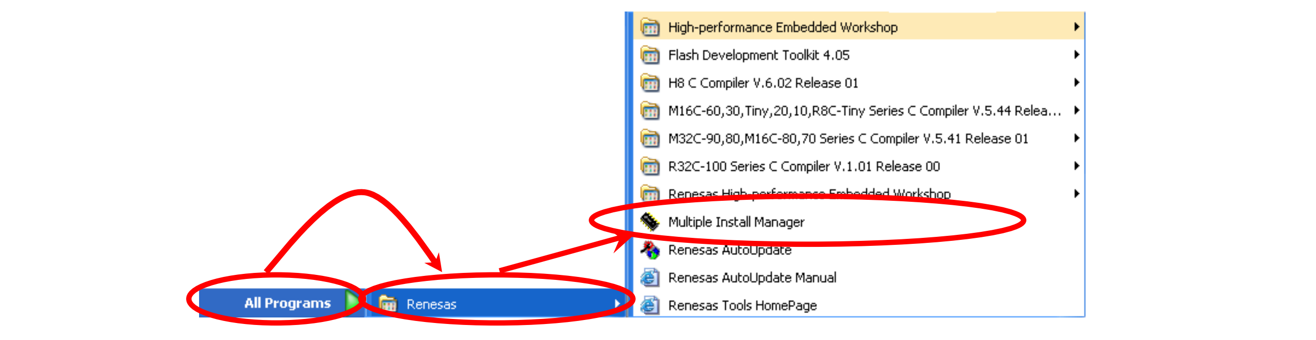
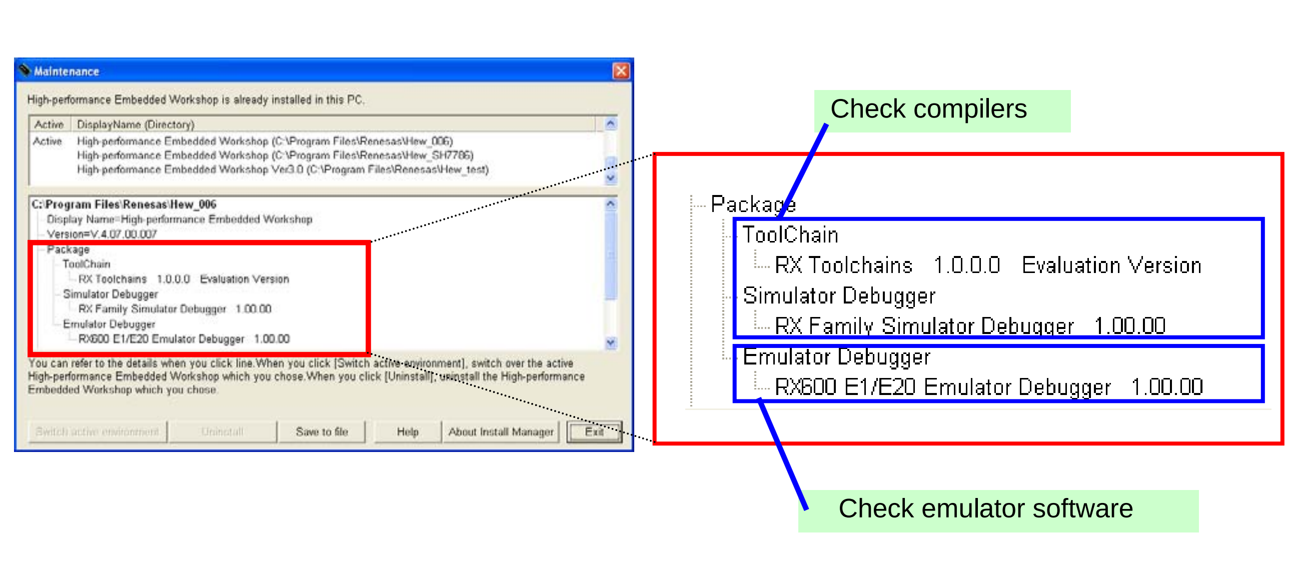

All Programs
Renesas
High-performance Embedded Workshop
Flash Development Toolkit 4.05
H8 C Compiler V.6.02 Release 01
M16C-60,30,Tiny,20,10,R8C-Tiny Series C Compiler V.5.44 Relea...
M32C-90,80,M16C-80,70 Series C Compiler V.5.41 Release 01
R32C-100 Series C Compiler V.1.01 Release 00
Renesas High-performance Embedded Workshop
Multiple Install Manager
Renesas AutoUpdate
Renesas AutoUpdate Manual
Renesas Tools HomePage

Figure 3.15 Starting up the Multiple Install Manager

(2) When [Multiple Install Manager] is started up, the [Maintenance] dialog box appears (Figure 3.16). Read the contents and check that all of the required products have been installed. If anything is missing, re-install it.



High-performance Embedded Workshop is already installed in this PC.

Active	DisplayName (Directory)
Active	High-performance Embedded Workshop (C:\Program Files\Renesas\Hew_006)
	High-performance Embedded Workshop (C:\Program Files\Renesas\Hew_SH7786)
	High-performance Embedded Workshop Ver3.0 (C:\Program Files\Renesas\Hew_test)

```
C:\Program Files\Renesas\Hew_006
  Display Name=High-performance Embedded Workshop
  Version=V.4.07.00.007
  Package
    ToolChain
      RX Toolchains   1.0.0.0   Evaluation Version
    Simulator Debugger
      RX Family Simulator Debugger   1.00.00
    Emulator Debugger
      RX600 E1/E20 Emulator Debugger   1.00.00
```

You can refer to the details when you click line.When you click [Switch active environment], switch over the active High-performance Embedded Workshop which you chose.When you click [Uninstall], uninstall the High-performance Embedded Workshop which you chose.

Figure 3.16 Checking the Products That Have been Installed

3.6 Checking That the USB Driver Has been Installed

To check that the USB driver is working correctly, see the state of the ACT LED on the E1 emulator after connecting the emulator and host computer via a USB interface cable. Table 3.1 shows the states of the ACT LED and their meanings.

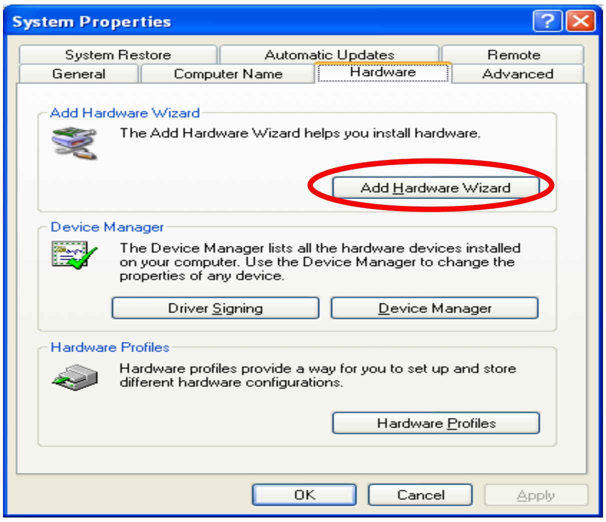
Table 3.1 ACT States and Meanings

ACT LED	Communication between the Host Computer and Emulator	Checkpoint
Illuminated	The E1 emulator is usable.	The USB driver has correctly been installed.
Blinking	The E1 emulator is not usable yet because the USB driver has not been recognized.	Check whether the USB driver has correctly been installed (by following the procedure below).
Not illuminated	Communication has not been established.	Check whether the USB interface cable is correctly connected, the host computer and emulator are supplied with power, and no items are damaged.

Follow the procedures below to check whether the USB driver has successfully been installed.

(1) Open the properties window for [My Computer] ([System Properties] dialog box).

(2) Open the [Hardware] tabbed page and click on the [Add Hardware Wizard] button as shown in Figure 3.17.



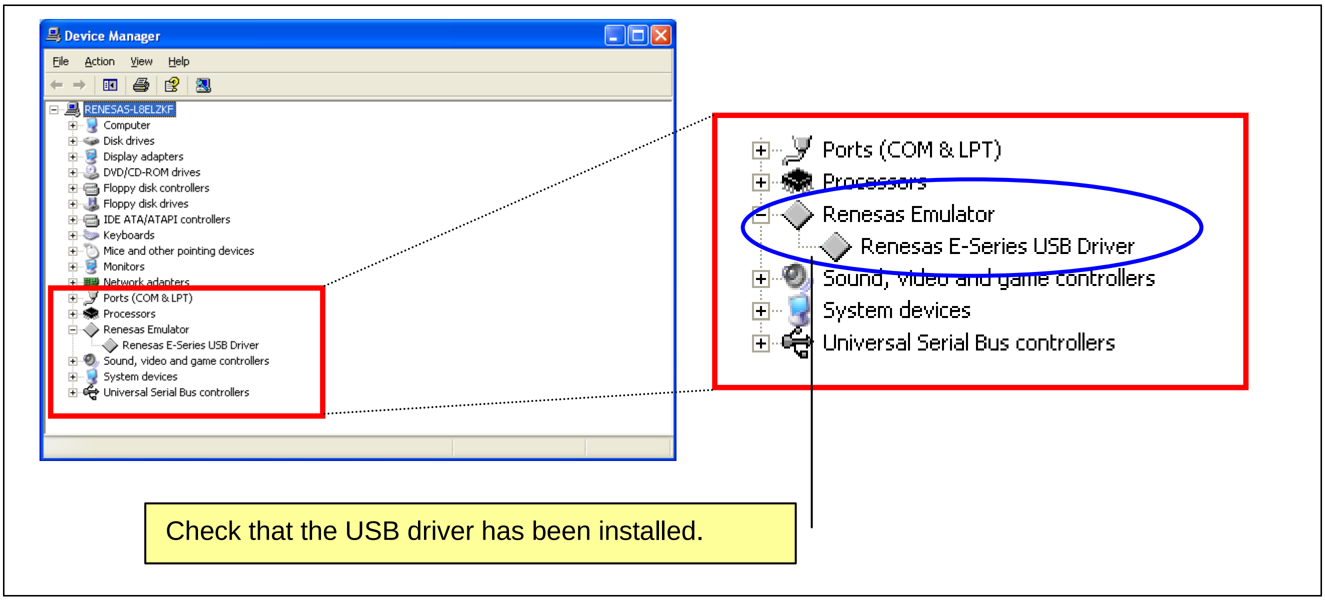
The Add Hardware Wizard helps you install hardware.

The Device Manager lists all the hardware devices installed on your computer. Use the Device Manager to change the properties of any device.

Hardware profiles provide a way for you to set up and store different hardware configurations.

Figure 3.17 [System Properties] Dialog Box

(3) Check that [Renesas E-Series USB Driver] is shown under the [Renesas Emulator] category (Figure 3.18).



Check that the USB driver has been installed.

Figure 3.18 Device Manager

Some drivers under the [Renesas Emulator] category may have a [!] or [?] mark attached to the icon. These drivers may not have been successfully installed, so delete and then re-install them by following the procedure given in section 3.1, Before Installation.

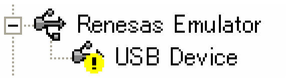


Figure 3.19 Display Indicating Failed Instllation of a USB Driver

If you cannot find any driver under the [Renesas Emulator] category, check that the emulator software has been installed and repeat the procedure given in section 3.4, Installing the USB Driver.

If you have found [Renesas E-Series USB Driver] under the [Renesas Emulator] category but the ACT LED is blinking (i.e. the E1 emulator is not usable), delete [Renesas E-Series USB Driver] and then re-install it.

4. Let's Try Using the E1 Emulator

This section describes the basic usage of the E1 emulator with regard to the tutorial program installed in the host computer during installation of the E1 emulator software.

4.1 Emulator Operating Procedure

This section describes the emulator operations (High-performance Embedded Workshop activation, tutorial program downloading, and program execution) and usage of the main debugging functions of the E1 emulator.

Table 4.1 shows the operating procedures (this assumes that the necessary installation is completed).

Table 4.1 Tutorial Program Execution Procedure

Step	Operation	Reference Section in This Guide
1	Connect the E1 emulator to the host computer through the USB interface cable	4.2 Connecting the Emulator
2	Connect the user system	
3	Activate the High-performance Embedded Workshop	4.3 Activating the High-performance Embedded Workshop and Loading the Tutorial
4	Open the workspace	
5	Set up the E1 emulator during booting up	4.4 Setting up the E1 Emulator
6	Configuration property	
7	Open a source file	4.5 Opening a Source File
8	Build (compile and link)	4.6 Building a Program (Compiling and Linking)
9	Download a program	4.7 Downloading a Program
10	Reset the CPU	4.8 Resetting the CPU
11	Start and stop the program	4.9 Starting and Stopping Program Execution
12	Break functions	4.10 Break Functions
13	Refer to or modify C-language variables	4.11 Referring to or Modifying C-Language Variables
14	Refer to or modify data through the memory window	4.12 Referring to or Modifying Data through the [Memory] Window
15	Refer to or modify registers	4.13 Referring to or Modifying Register Values through the [Register] Window
16	Refer to or modify I/O registers	4.14 Referring to or Modifying I/O Register Values through the [I/O] Window
17	Analyze the performance	4.15 Analyzing Performance
18	Trace function	4.16 Trace Function
19	Terminate debugging	4.17 Terminating Debugging
20	Operate the user board as a stand-alone unit	4.18 Operating the User Board as a Stand-Alone Unit

4.2 Connecting the Emulator

Figure 4.1 shows the configuration of the E1 emulator system.

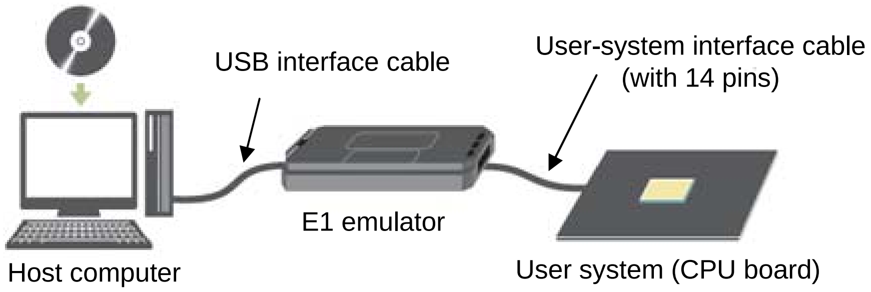


Figure 4.1 System Configuration with the Emulator

Connect the E1 emulator system through the following steps.

(1) Connect the E1 emulator to the host computer through the USB interface cable.
(2) Connect the E1 emulator to the user system (CPU board) through the user-system interface cable.

Note: When using a separate power supply for the user system, be sure to check that the power is not supplied before connecting the E1 emulator.

4.3 Activating the High-performance Embedded Workshop and Loading the Tutorial

Activate the High-performance Embedded Workshop and open the tutorial workspace through the following steps.

(1) From [All Programs] in the [Start] menu, select [Renesas -> High-performance Embedded Workshop -> High-performance Embedded Workshop] as shown in Figure 4.2 to activate the High-performance Embedded Workshop.

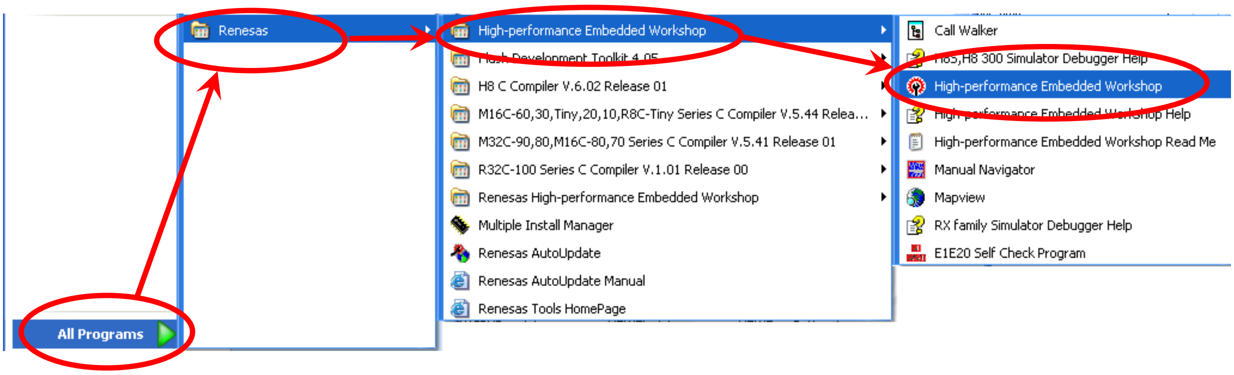

All Programs
Renesas
High-performance Embedded Workshop
Flash Development Toolkit 4.05
H8 C Compiler V.6.02 Release 01
M16C-60,30,Tiny,20,10,R8C-Tiny Series C Compiler V.5.44 Relea...
M32C-90,80,M16C-80,70 Series C Compiler V.5.41 Release 01
R32C-100 Series C Compiler V.1.01 Release 00
Renesas High-performance Embedded Workshop
Multiple Install Manager
Renesas AutoUpdate
Renesas AutoUpdate Manual
Renesas Tools HomePage
Call Walker
H8S,H8 300 Simulator Debugger Help
High-performance Embedded Workshop
High-performance Embedded Workshop Help
High-performance Embedded Workshop Read Me
Manual Navigator
Mapview
RX family Simulator Debugger Help
E1E20 Self Check Program

Figure 4.2 Activating the High-performance Embedded Workshop

(2) The [Welcome!] dialog box will appear on the High-performance Embedded Workshop screen as shown in Figure 4.3. Select the [Browse to another project workspace] radio button and click on the [OK] button.

(3) The [Open Workspace] dialog box shown in Figure 4.3 will appear.
Specify "Tutorial.hws" and click on the [Select] button.
When the software of this product has been successfully installed, workspace "Tutorial.hws" is stored in the folder shown below.

<Drive where the OS is installed>:
`\WorkSpace\Tutorial\E1E20\RX600\Tutorial_LittleEndian`

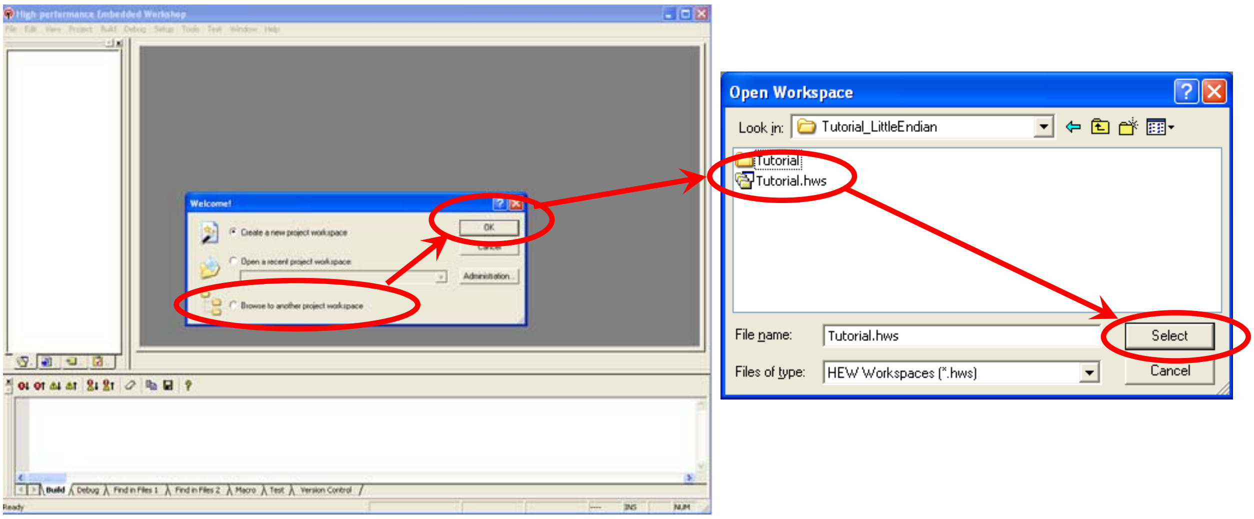
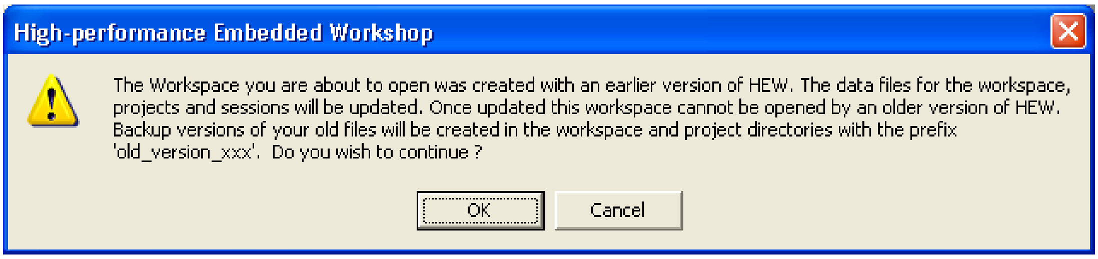

High-performance Embedded Workshop
Welcome!
○ Create a new project workspace
○ Open a recent project workspace
○ Browse to another project workspace
OK
Cancel
Administration...
Open Workspace
Look in: Tutorial_LittleEndian
Tutorial
Tutorial.hws
File name: Tutorial.hws
Files of type: HEW Workspaces (*.hws)
Select
Cancel

Figure 4.3 Selecting a Workspace in the [Welcome!] Dialog Box

If the version of the High-performance Embedded Workshop used for creating the installed tutorial workspace is older than what is currently used in your computer, the dialog box shown in Figure 4.4 will appear.

In this case, to update the workspace version to the current High-performance Embedded Workshop version, click on the [OK] button.

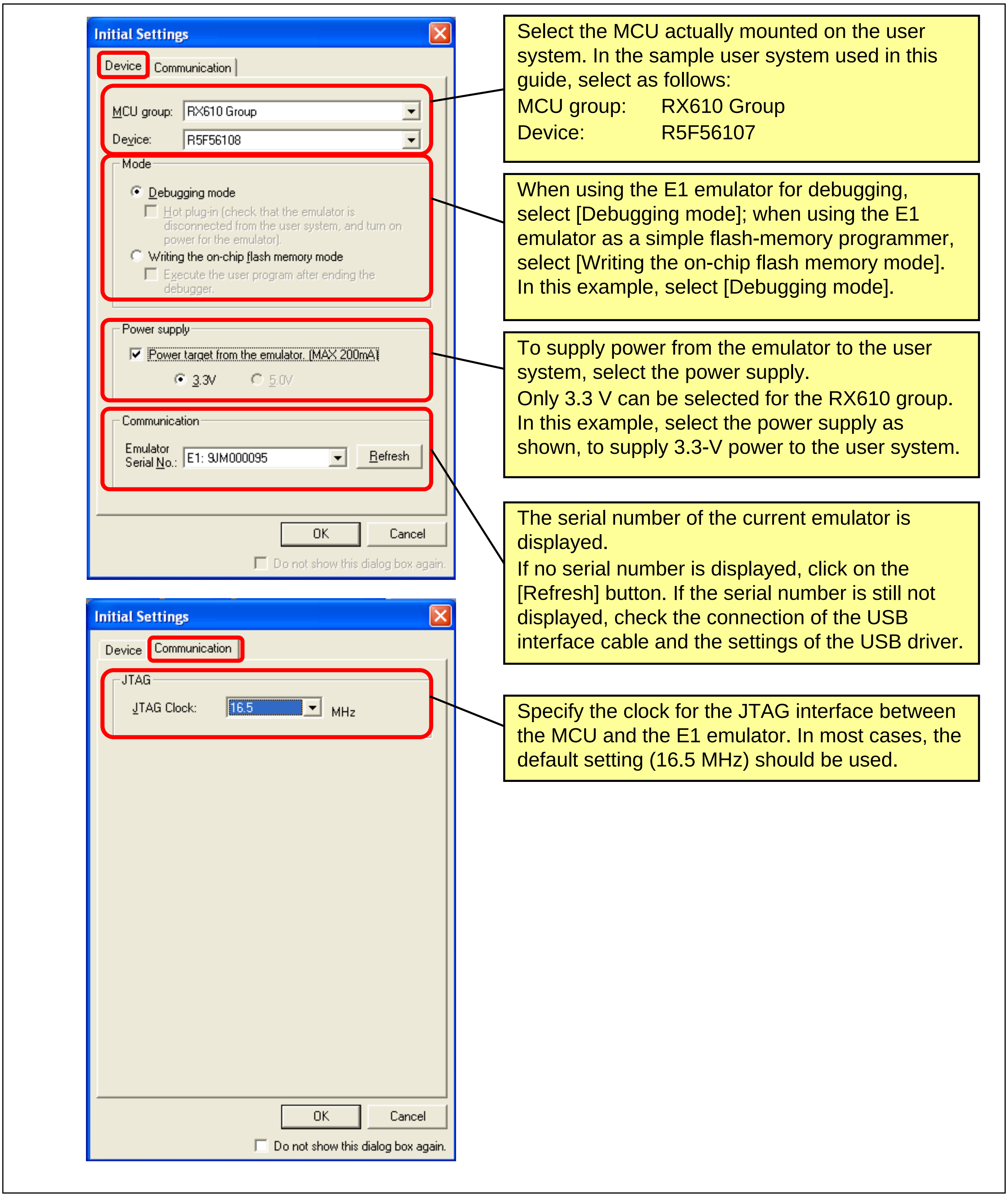

High-performance Embedded Workshop

The Workspace you are about to open was created with an earlier version of HEW. The data files for the workspace, projects and sessions will be updated. Once updated this workspace cannot be opened by an older version of HEW. Backup versions of your old files will be created in the workspace and project directories with the prefix 'old_version_xxx'. Do you wish to continue ?

Figure 4.4 Dialog Box Shown for Old-Version Workspace

4.4 Setting up the E1 Emulator during Booting up

(1) The [Initial Settings] dialog box shown in Figure 4.5 will appear. Make the necessary settings as shown in the figure and click on the [OK] button.
When the [Initial Settings] dialog box does not appear, switch the session and click on the [Connect] button as shown in Figure 4.6.



Initial Settings

Device | Communication

MCU group: RX610 Group

Device: R5F56108

Mode

● Debugging mode

□ Hot plug-in (check that the emulator is disconnected from the user system, and turn on power for the emulator).

○ Writing the on-chip flash memory mode

□ Execute the user program after ending the debugger.

Power supply

☑ Power target from the emulator. (MAX 200mA)

● 3.3V ○ 5.0V

Communication

Emulator Serial No.: E1: 9JM000095

Initial Settings

Device | Communication

JTAG

JTAG Clock: 16.5 MHz

Select the MCU actually mounted on the user system. In the sample user system used in this guide, select as follows:
MCU group: RX610 Group
Device: R5F56107

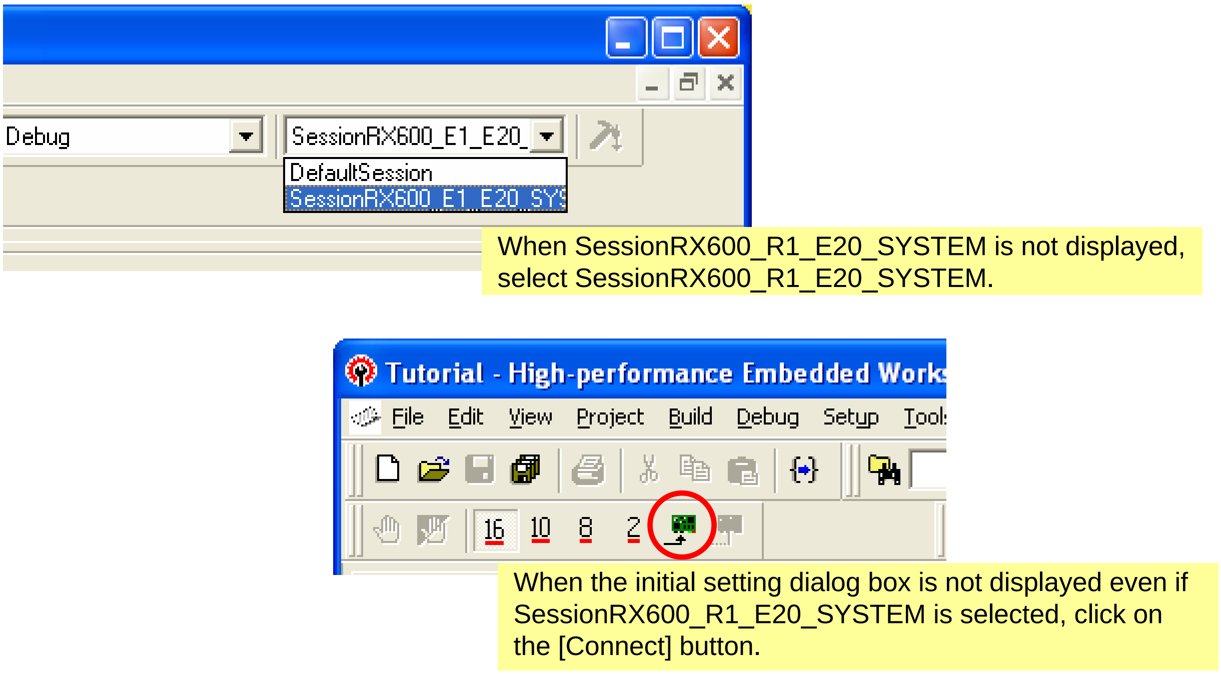
When using the E1 emulator for debugging, select [Debugging mode]; when using the E1 emulator as a simple flash-memory programmer, select [Writing the on-chip flash memory mode]. In this example, select [Debugging mode].

To supply power from the emulator to the user system, select the power supply.
Only 3.3 V can be selected for the RX610 group. In this example, select the power supply as shown, to supply 3.3-V power to the user system.

The serial number of the current emulator is displayed.
If no serial number is displayed, click on the [Refresh] button. If the serial number is still not displayed, check the connection of the USB interface cable and the settings of the USB driver.

Specify the clock for the JTAG interface between the MCU and the E1 emulator. In most cases, the default setting (16.5 MHz) should be used.

Figure 4.5 Initial Settings

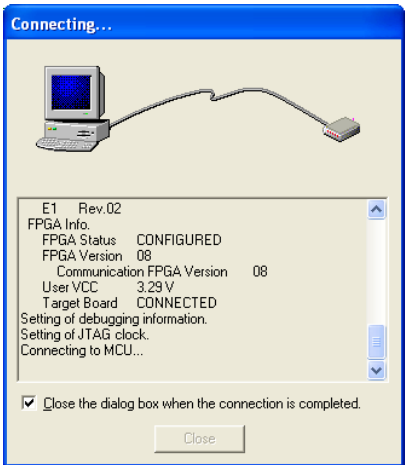

Debug
SessionRX600_E1_E20_
DefaultSession
SessionRX600_E1_E20_SYS
Tutorial - High-performance Embedded Work
File Edit View Project Build Debug Setup Tools
16 10 8 2

When SessionRX600_R1_E20_SYSTEM is not displayed, select SessionRX600_R1_E20_SYSTEM.

When the initial setting dialog box is not displayed even if SessionRX600_R1_E20_SYSTEM is selected, click on the [Connect] button.

Figure 4.6　Session Switching and [Connect] Button

While the connection processing is in progress, the [Connecting] dialog box shown in Figure 4.7 is displayed.

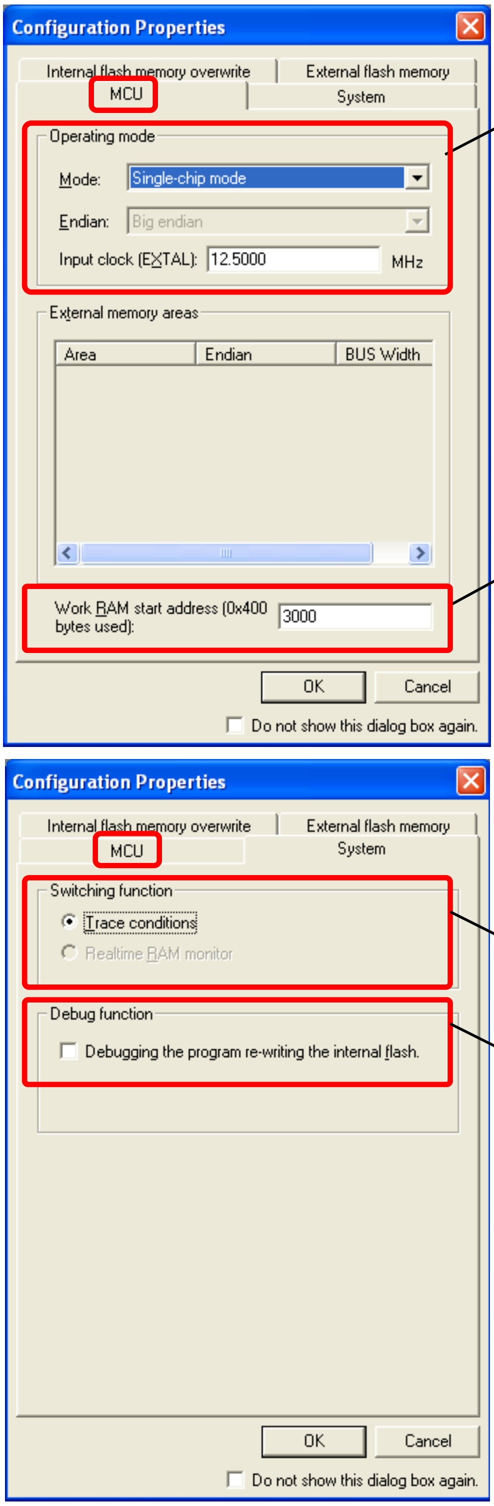


```
   E1     Rev.02
FPGA Info.
   FPGA Status     CONFIGURED
   FPGA Version    08
      Communication FPGA Version      08
   User VCC        3.29 V
   Target Board    CONNECTED
Setting of debugging information.
Setting of JTAG clock.
Connecting to MCU...
```

☑ Close the dialog box when the connection is completed.

Figure 4.7　Dialog Box Shown during Emulator Connection

(2) The [Configuration Properties] dialog box shown in Figure 4.8 will appear.
Check that the options in the [MCU] and [System] tabs are set as shown in Figure 4.8 and click on the [OK] button.



Operating mode

Mode: Single-chip mode

Endian: Big endian

Input clock (EXTAL): 12.5000 MHz

External memory areas

Area	Endian	BUS Width

Work RAM start address (0x400 bytes used): 3000

□ Do not show this dialog box again.

Switching function

● Trace conditions

○ Realtime RAM monitor

Debug function

□ Debugging the program re-writing the internal flash.

□ Do not show this dialog box again.

Mode:
Select the MCU operating mode used for the target program to be debugged.
In this example, select the single-chip mode.
Endian:
The endian specified in the user system is automatically displayed.
Input clock:
Specify the frequency of the clock signal connected to the XTAL pin of the MCU.
In the RX610 CPU board used for this example, a 12.5-MHz clock signal is connected.

Specify the work RAM area to be used for the emulator.
In most cases, the default setting should be used. However, when using the default area as the DMA or DTC transfer source or destination, specify another area.
In this example, use the default setting.

The E1 emulator does not support the realtime RAM monitoring function, and the radio button for the function cannot be selected.

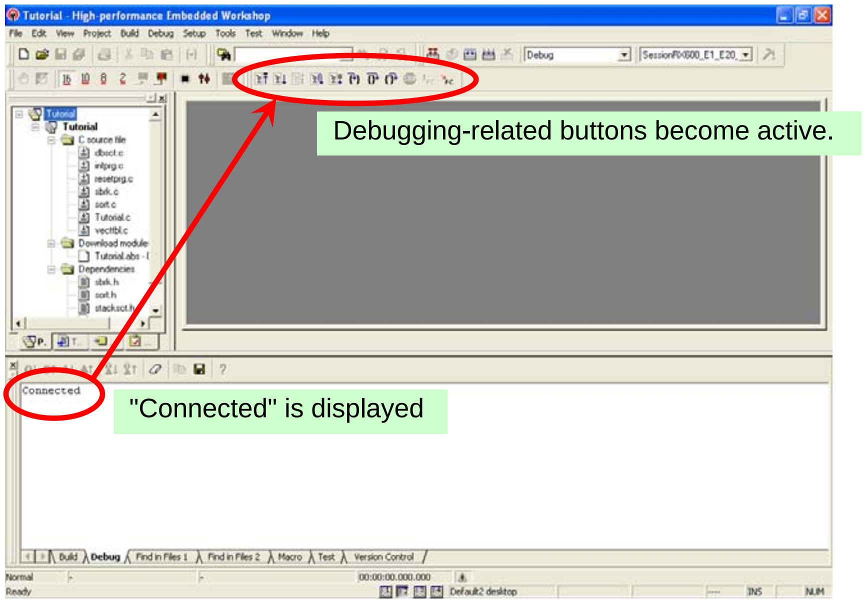
Select this function when debugging a program that erases or overwrites the on-chip flash memory contents.
The tutorial program does not overwrite the on-chip flash memory contents; do not select this function in this example.

The other options should be left as the default settings. For details of the other options, refer to the E1/E20 Emulator User's Manual.

Figure 4.8 [Configuration Properties] Dialog Box

(3) The [Connecting] dialog box will appear.

(4) When the emulator has been connected successfully, "Connected" will be displayed in the output window and the debugging-related buttons in the upper toolbar will become active as shown in Figure 4.9.



```
Connected
```

Debugging-related buttons become active.

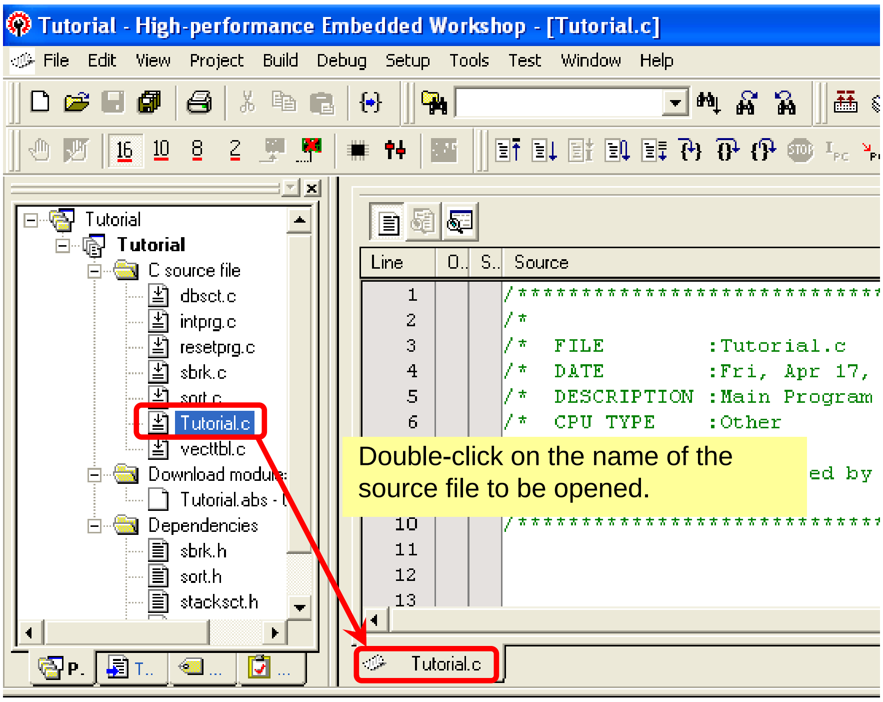
"Connected" is displayed

Figure 4.9 Confirming Emulator Connection

4.5 Opening a Source File

To open a source fie, double-click on the source file name in the workspace window. For example, to open Tutorial.c, double-click on the file name as shown in Figure 4.10.

The window for displaying the source code will open; it is called the editor window because the source code can be modified in this window.



```
/*****************************
/*
/*  FILE        :Tutorial.c
/*  DATE        :Fri, Apr 17,
/*  DESCRIPTION :Main Program
/*  CPU TYPE    :Other
                          ed by
/*****************************
```

Double-click on the name of the source file to be opened.

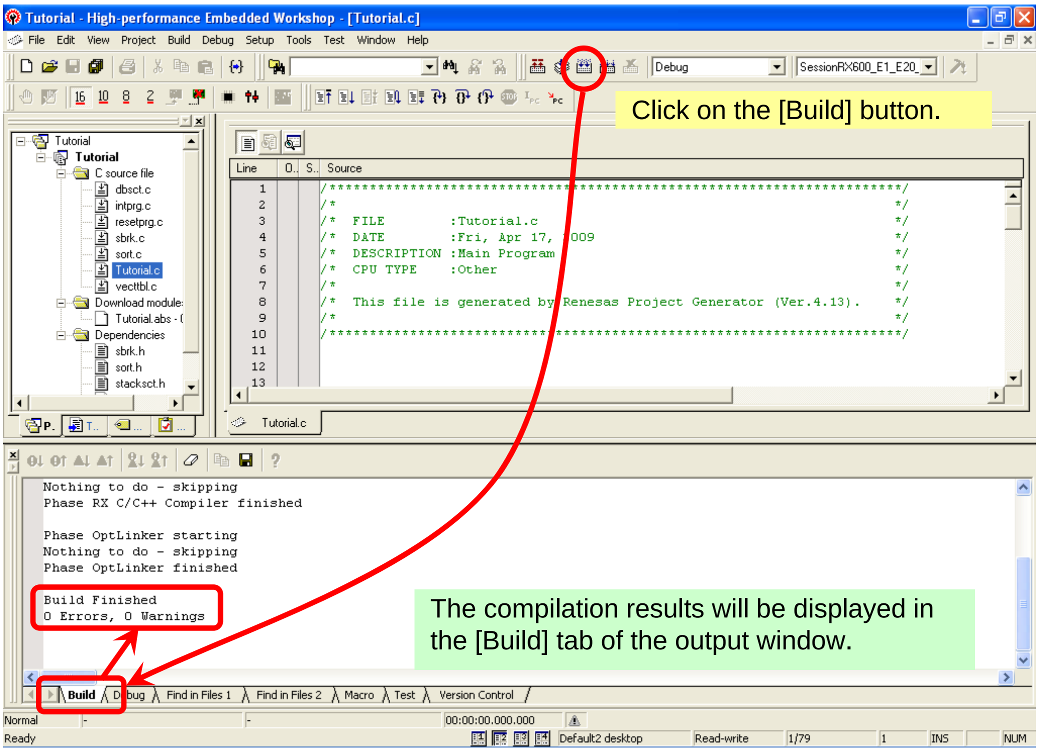
Figure 4.10 Opening a Source File

4.6 Building a Program (Compiling and Linking)

To build a program, click on the [Build] button on the toolbar.

When the build process has been completed, the build results are displayed in the output window. In this tutorial program, some warning messages will appear but ignore them.

Note: When the build process is done for the first time, the standard library is also compiled and the process takes some time. In the second or later time, compiling the standard library is not necessary and the processing time will be reduced unless any option is modified.



```
/*************************************************************************/
/*                                                                       */
/*   FILE        :Tutorial.c                                             */
/*   DATE        :Fri, Apr 17, 2009                                      */
/*   DESCRIPTION :Main Program                                           */
/*   CPU TYPE    :Other                                                  */
/*                                                                       */
/*   This file is generated by Renesas Project Generator (Ver.4.13).    */
/*                                                                       */
/*************************************************************************/
```

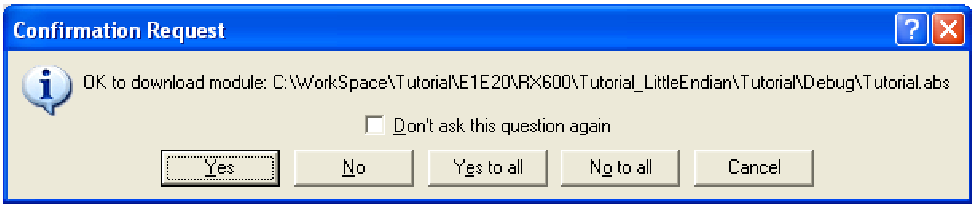
```
Nothing to do - skipping
Phase RX C/C++ Compiler finished

Phase OptLinker starting
Nothing to do - skipping
Phase OptLinker finished

Build Finished
0 Errors, 0 Warnings
```

Click on the [Build] button.

The compilation results will be displayed in the [Build] tab of the output window.

Figure 4.11 Build Execution and Results

The workspace for the tutorial program is set up so that the program is automatically downloaded after the build process. When the dialog box shown in Figure 4.12 appears, click on the [Yes] button to start downloading the program.



OK to download module: C:\WorkSpace\Tutorial\E1E20\RX600\Tutorial_LittleEndian\Tutorial\Debug\Tutorial.abs

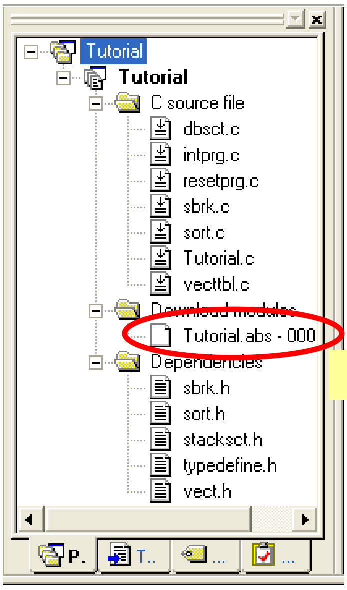
□ Don't ask this question again

Figure 4.12 Confirmation of Downloading

4.7 Downloading a Program

In addition to using the automatic downloading function after a build process, a program can be downloaded through the following procedure.

(1) Double-click on the load module file name ("Tutorial.abs" in this example) displayed under [Download module] in the workspace window as shown in Figure 4.13.



Double-click on the file name.

Figure 4.13 Downloading a Program

(2) When loading is completed, a down arrow is added on the file icon as shown in Figure 4.14.

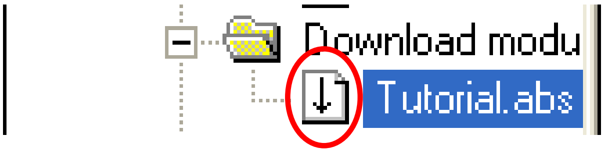


Figure 4.14 Program Downloading Completed

4.8 Resetting the CPU

To reset the CPU, select [Reset CPU] from the [Debug] menu or click on the [Reset CPU] button on the toolbar.

This will reset the registers to the initial values and set the program counter to the reset vector address

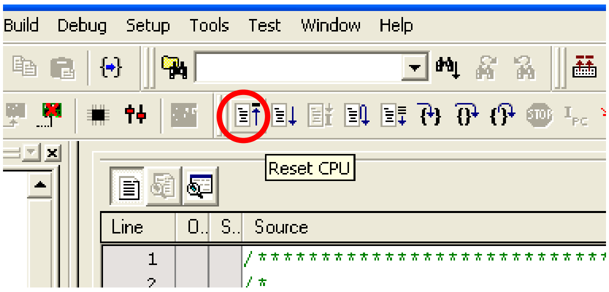
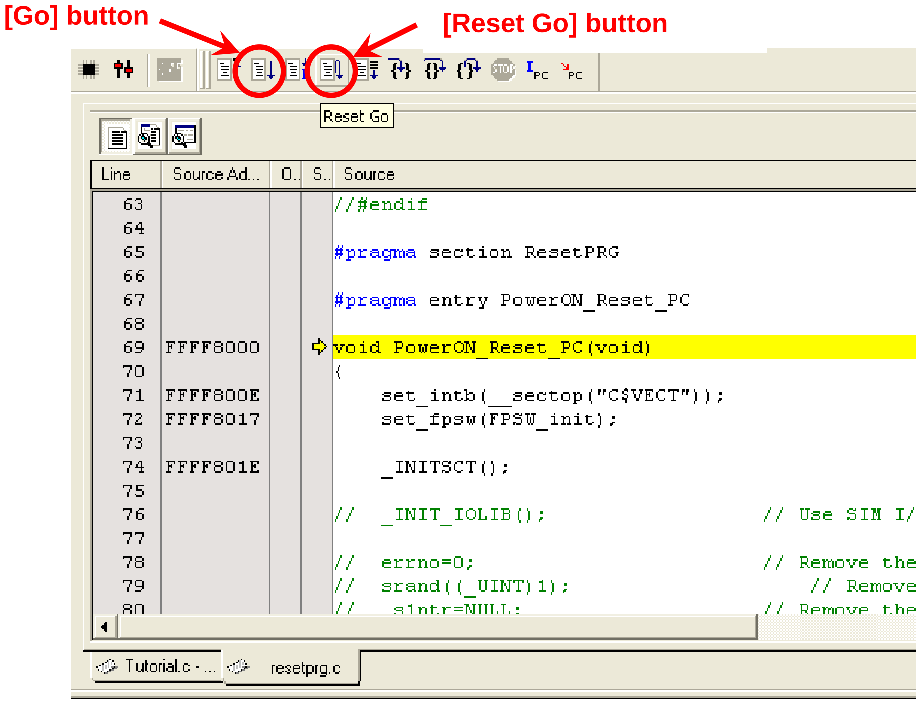


Figure 4.15 [Reset CPU] Button

4.9 Starting and Stopping Program Execution

Click on the [Go] button shown in Figure 4.16 to start program execution.

The [Reset Go] button is also available to start program execution after resetting the CPU.



```
           //#endif

           #pragma section ResetPRG

           #pragma entry PowerON_Reset_PC

FFFF8000   void PowerON_Reset_PC(void)
           {
FFFF800E       set_intb(__sectop("C$VECT"));
FFFF8017       set_fpsw(FPSW_init);

FFFF801E       _INITSCT();

           //  _INIT_IOLIB();                // Use SIM I/

           //  errno=0;                      // Remove the
           //  srand((_UINT)1);                  // Remove
           //   _s1ptr=NULL;                 // Remove the
```

Figure 4.16 [Go] Buttons

The [Halt] button shown in Figure 4.17 is active during program execution.

Click on this button to stop program execution.

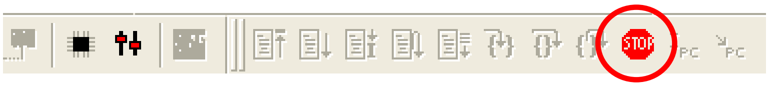
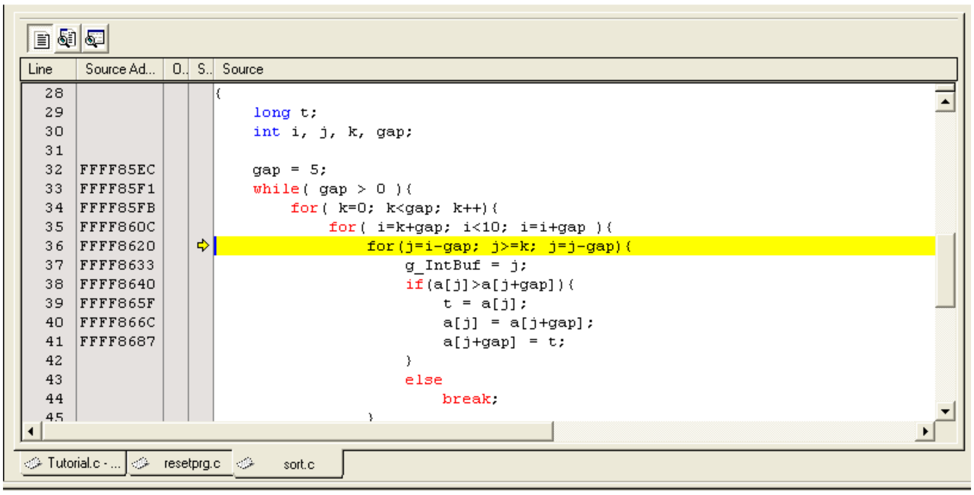


Figure 4.17 [Halt] Button

Figure 4.18 shows the editor window after program execution stops.

The line highlighted in yellow corresponds to the program counter address when execution stops.



```
          {
              long t;
              int i, j, k, gap;

FFFF85EC      gap = 5;
FFFF85F1      while( gap > 0 ){
FFFF85FB          for( k=0; k<gap; k++){
FFFF860C              for( i=k+gap; i<10; i=i+gap ){
FFFF8620                  for(j=i-gap; j>=k; j=j-gap){
FFFF8633                      g_IntBuf = j;
FFFF8640                      if(a[j]>a[j+gap]){
FFFF865F                          t = a[j];
FFFF866C                          a[j] = a[j+gap];
FFFF8687                          a[j+gap] = t;
                              }
                              else
                                  break;
                  }
```

Figure 4.18 Editor Window after Program Execution Stops

4.10 Break Functions

The E1 emulator provides two types of break function: software break function and on-chip break function.

Table 4.2 shows the characteristics of the software break and on-chip break functions. Please refer to the table to choose break functions suitable for each debugging purpose.

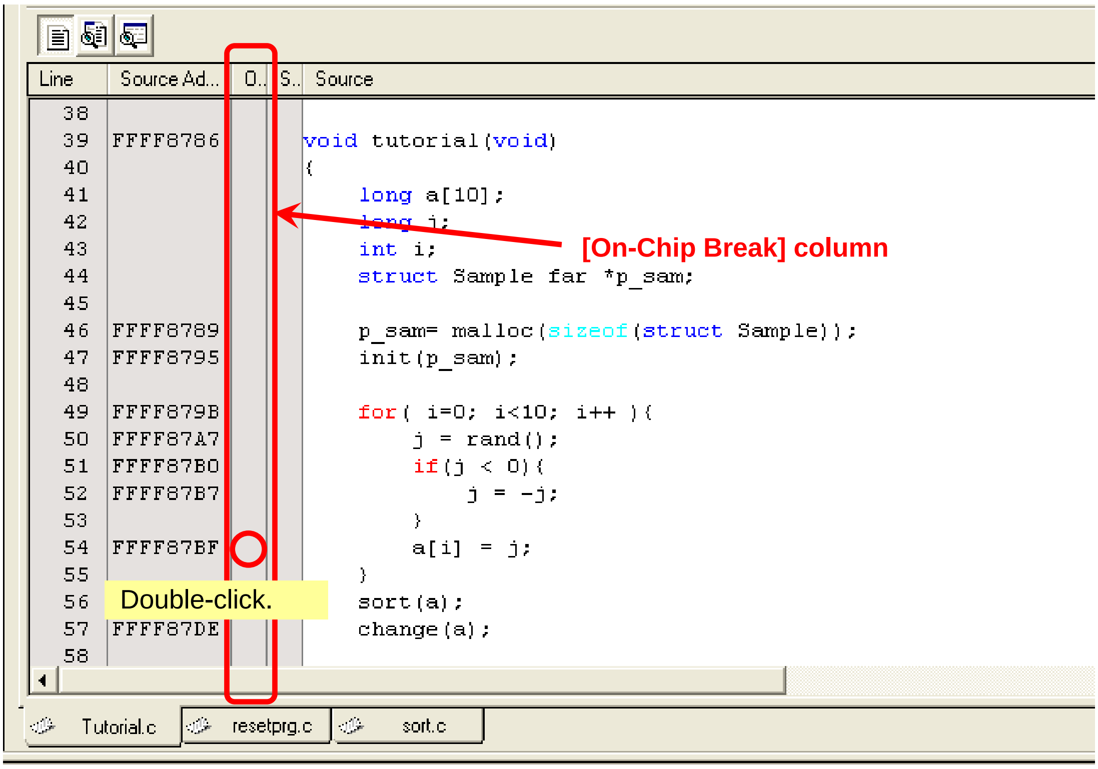
Table 4.2 Break Function Specifications

Break Function	Specifiable Points	Break Condition	Flash Memory Overwriting	Availability During Program Execution
Software break				
	256 points	Specified address	Overwritten	Specifiable (only in the on-chip RAM area)
On-chip break				
Pre-PC break	8 points	Specified address	None	Specifiable
Event break (execution address)		Specified address	None	Specifiable
Event break (data access)	4 points	Data access	None	Specifiable
Trace full break	-	Trace buffer becomes full	None	Not specifiable

Software break can be specified at many points, but this function is implemented by modifying the contents of the specified addresses to dedicated break instructions; extra time is needed to modify the contents during program execution after breakpoints are specified. Accordingly, the on-chip break function is recommended in most cases and the software break function should be used only for the addresses where breakpoints will not be deleted and reset repeatedly.

This section describes the procedure for specifying the on-chip break function, which does not require flash-memory overwriting.

(1) Set a breakpoint.
Double-click on the on-chip break column shown in Figure 4.19 and a blue-filled circle will appear.
A breakpoint can be specified only for a line where an address is shown in the [Source Address] column.



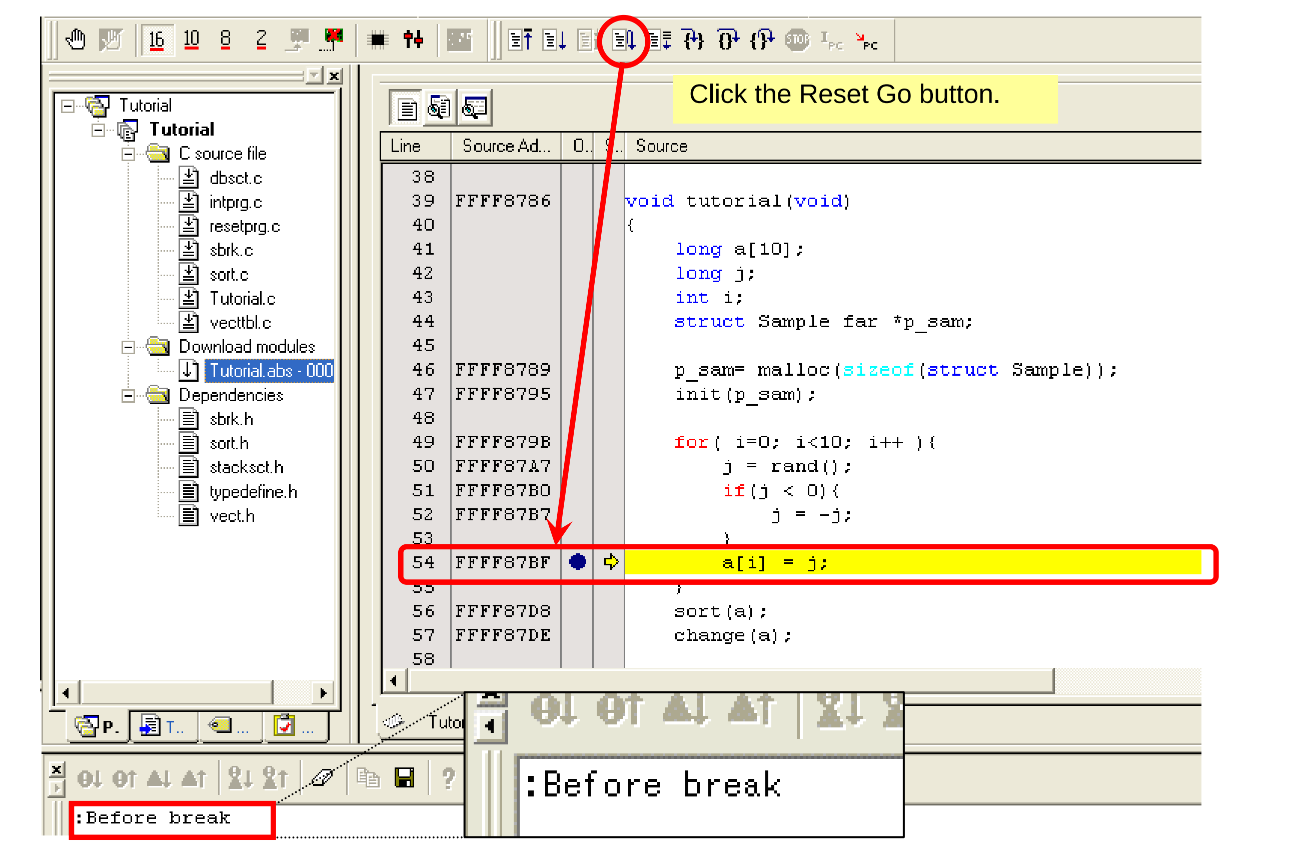
```
FFFF8786   void tutorial(void)
           {
               long a[10];
               long j;
               int i;
               struct Sample far *p_sam;

FFFF8789       p_sam= malloc(sizeof(struct Sample));
FFFF8795       init(p_sam);

FFFF879B       for( i=0; i<10; i++ ){
FFFF87A7           j = rand();
FFFF87B0           if(j < 0){
FFFF87B7               j = -j;
                   }
FFFF87BF           a[i] = j;
               }
               sort(a);
FFFF87DE       change(a);
```

[On-Chip Break] column

Double-click.

Figure 4.19 Setting a Breakpoint

(2) Click the [Reset Go] button as shown in Figure 4.20 to start program execution.
Program execution will stop at the breakpoint specified in the previous step.
The line corresponding to the program counter address where execution stops is highlighted in yellow as shown in Figure 4.20 and ":Before break" is displayed in the output window, which indicates that execution has stopped due to a break.

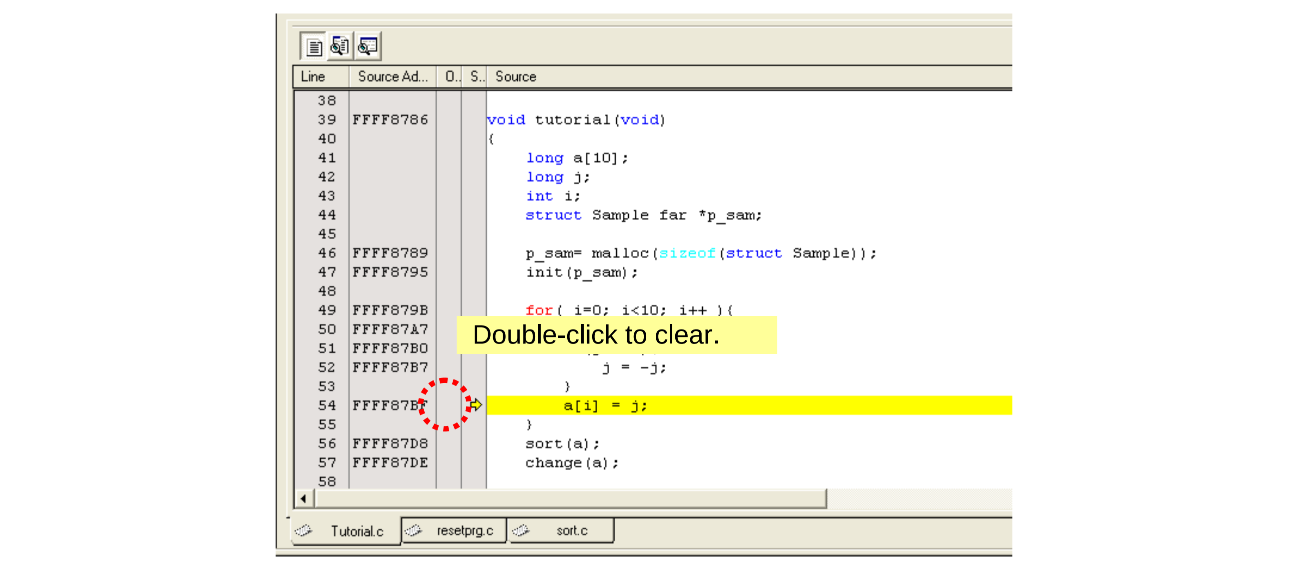


```
FFFF8786     void tutorial(void)
             {
                 long a[10];
                 long j;
                 int i;
                 struct Sample far *p_sam;

FFFF8789         p_sam= malloc(sizeof(struct Sample));
FFFF8795         init(p_sam);

FFFF879B         for( i=0; i<10; i++ ){
FFFF87A7             j = rand();
FFFF87B0             if(j < 0){
FFFF87B7                 j = -j;
                     }
FFFF87BF ●  ⇨           a[i] = j;
                 }
FFFF87D8         sort(a);
FFFF87DE         change(a);
```

```
:Before break
```

Figure 4.20 Execution Stopped at a Breakpoint

(3) To clear a breakpoint setting, double-click on the blue-filled circle as shown in Figure 4.21.



```
FFFF8786     void tutorial(void)
             {
                 long a[10];
                 long j;
                 int i;
                 struct Sample far *p_sam;

FFFF8789         p_sam= malloc(sizeof(struct Sample));
FFFF8795         init(p_sam);

FFFF879B         for( i=0; i<10; i++ ){
FFFF87A7
FFFF87B0
FFFF87B7                 j = -j;
                     }
FFFF87BF    ⇨           a[i] = j;
                 }
FFFF87D8         sort(a);
FFFF87DE         change(a);
```

Figure 4.21 Clearing a Breakpoint Setting

4.11 Referring to or Modifying C-Language Variables

The emulator provides the following functions for referring to or modifying C-language variables.

1. [Watch] window for referring to or modifying variables
   The values of any variables used in a program can be referred to or modified through this function. This function is available while program execution is in progress as well as when execution is stopped.
2. [Local] window for referring to or modifying variables
   The local variables used in the function pointed to by the current PC value are automatically displayed and can be modified.
3. Tooltip watch function for referring to variables
   The value of a variable in the source file displayed in a window can be easily checked by simply placing the cursor on the displayed variable name.

4.11.1 Using the [Watch] Window to Refer to or Modify Variables

The E1 emulator provides the [Watch] window, through which the values of any variables used in a program can be monitored.

This section describes how to use the watch function to check the changes in the value of global variable "int-type g_IntBuf" used in sample program sort.c

Note: When a local variable is specified to be watched, its value can be referred to only while the current PC address is within the function that has declared the local variable.

(1) Use the [Watch] window to refer to variables.
Select [View -> Symbol -> Watch] as shown in Figure 4.22 to open the [Watch] window.

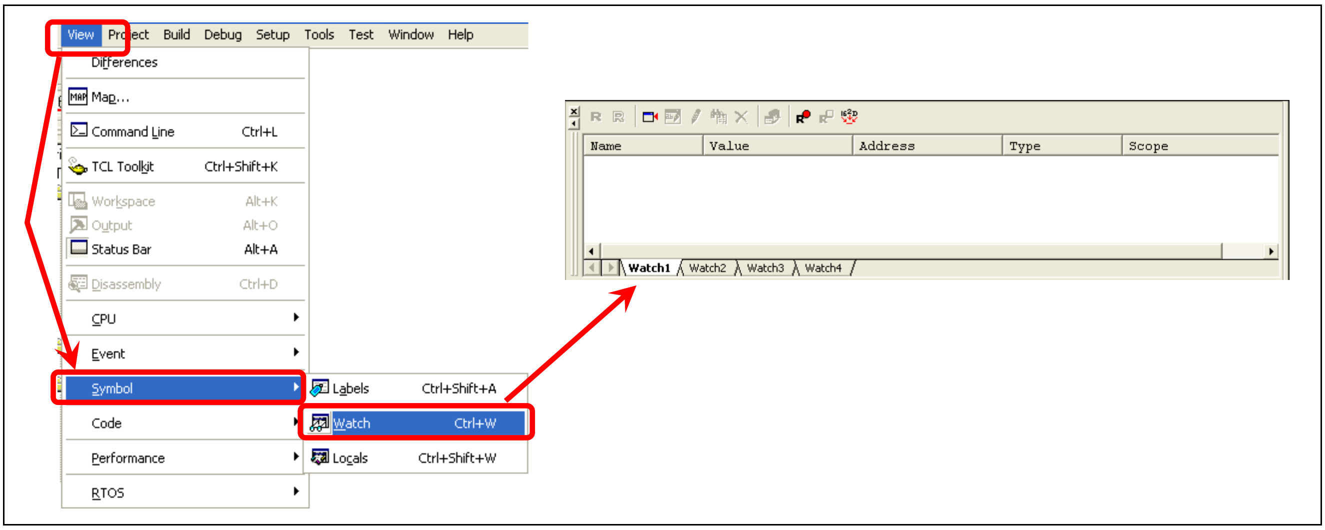
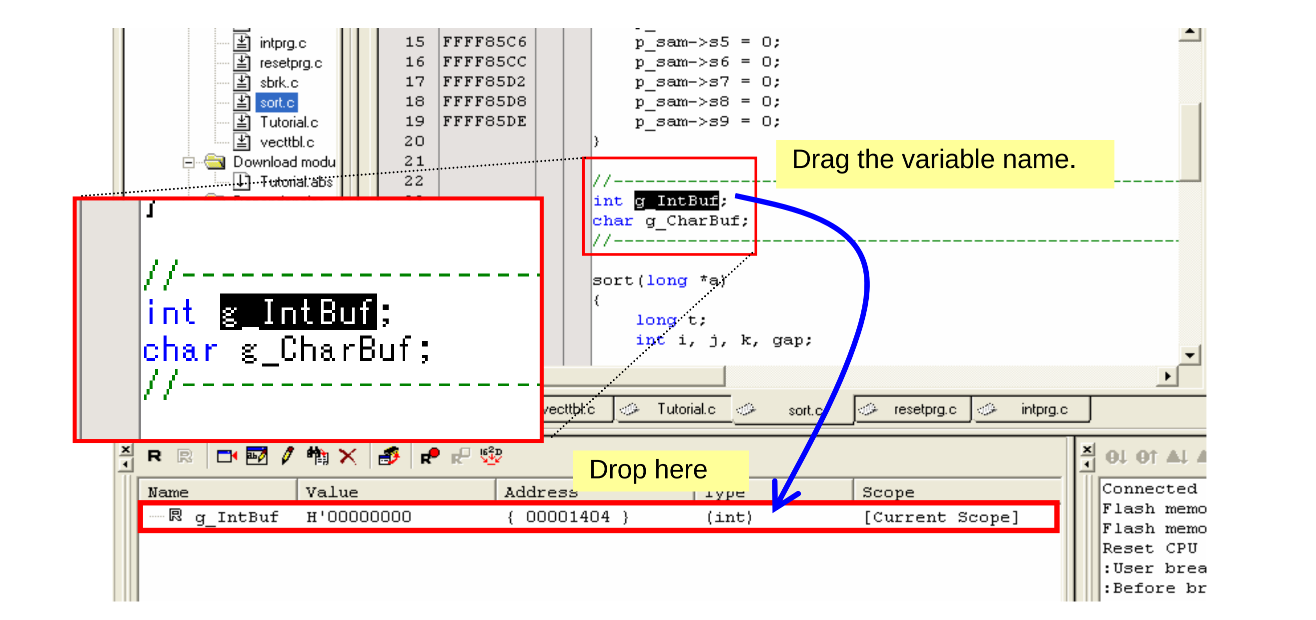


Figure 4.22 [Watch] Window

(2) Select g_IntBuf and then drag and drop it into the [Watch] window. Check that g_IntBuf is added to the [Watch] window as shown in Figure 4.23.

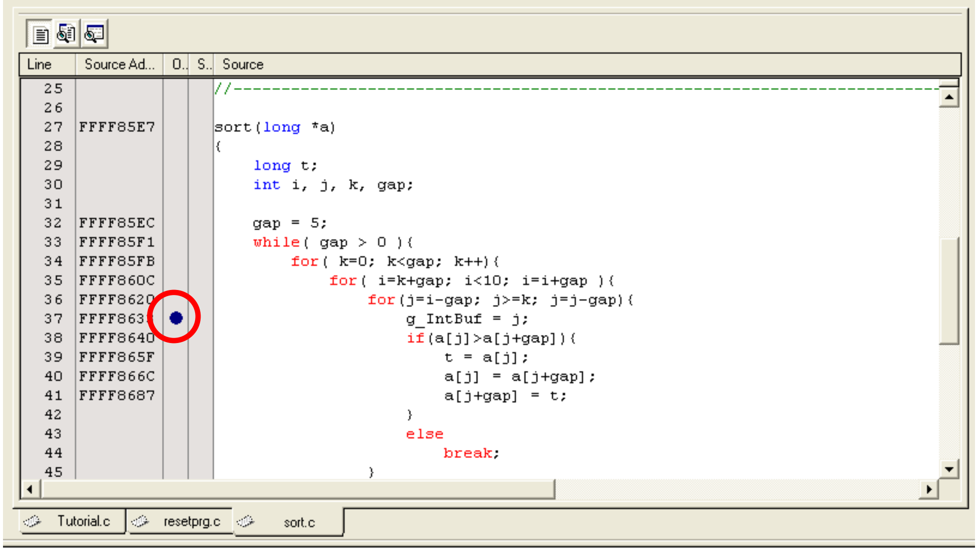


```
15  FFFF85C6      p_sam->s5 = 0;
16  FFFF85CC      p_sam->s6 = 0;
17  FFFF85D2      p_sam->s7 = 0;
18  FFFF85D8      p_sam->s8 = 0;
19  FFFF85DE      p_sam->s9 = 0;
20            }
21
22            //-----------------------------------------
              int g_IntBuf;
              char g_CharBuf;
              //-----------------------------------------

              sort(long *a)
              {
                  long t;
                  int i, j, k, gap;
```

Name	Value	Address	Type	Scope
g_IntBuf	H'00000000	{ 00001404 }	(int)	[Current Scope]

Figure 4.23 Watch Setting

(3) As shown in Figure 4.24, set an on-chip breakpoint at line 37 in sort.c, where "g_IntBuf = j;" is written. For details of the breakpoint setting procedure, refer to section 4.10, Break Functions.

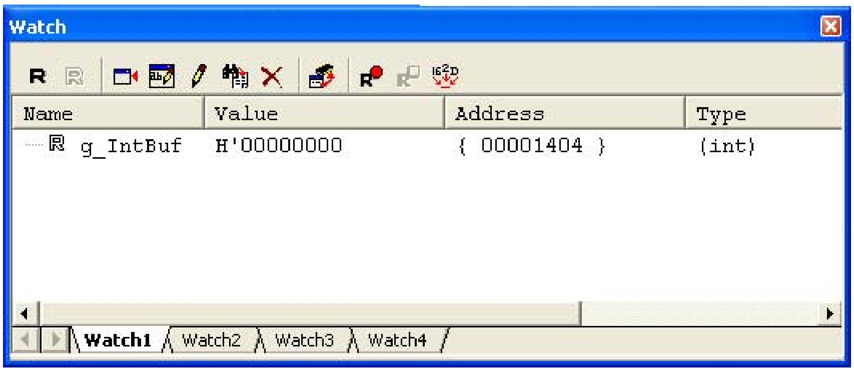


```
Line  Source Ad...  O.. S..  Source
 25                          //------------------------------------------------------------
 26
 27   FFFF85E7               sort(long *a)
 28                          {
 29                              long t;
 30                              int i, j, k, gap;
 31
 32   FFFF85EC                   gap = 5;
 33   FFFF85F1                   while( gap > 0 ){
 34   FFFF85FB                       for( k=0; k<gap; k++){
 35   FFFF860C                           for( i=k+gap; i<10; i=i+gap ){
 36   FFFF8620                               for(j=i-gap; j>=k; j=j-gap){
 37   FFFF863    ●                              g_IntBuf = j;
 38   FFFF8640                                   if(a[j]>a[j+gap]){
 39   FFFF865F                                       t = a[j];
 40   FFFF866C                                       a[j] = a[j+gap];
 41   FFFF8687                                       a[j+gap] = t;
 42                                              }
 43                                              else
 44                                                  break;
 45                                          }
```

Figure 4.24 Setting a Breakpoint

(4) After the above settings are completed, click on the [Reset Go] button to start program execution.

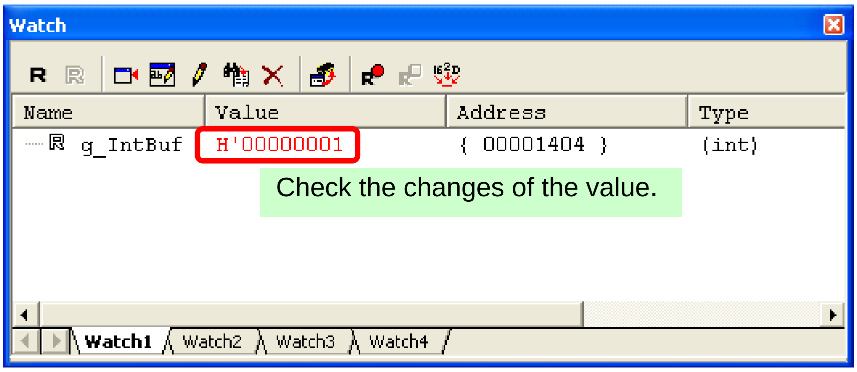
(5) At the first break, the program will stop before the specified line is executed and g_IntBuf will hold 0, which is the initial value.



Name	Value	Address	Type
g_IntBuf	H'00000000	{ 00001404 }	(int)

Figure 4.25 g_IntBuf Value after the First Break

(6) Click on the [Go] button instead of the [Reset Go] button to start program execution again; execution will break at the same point. After the break, check the change of the g_IntBuf value through the [Watch] window. Repeat these steps and check that the g_IntBuf value is incremented from the initial value: 0 -> 1 -> 2 -> 3 -> ...

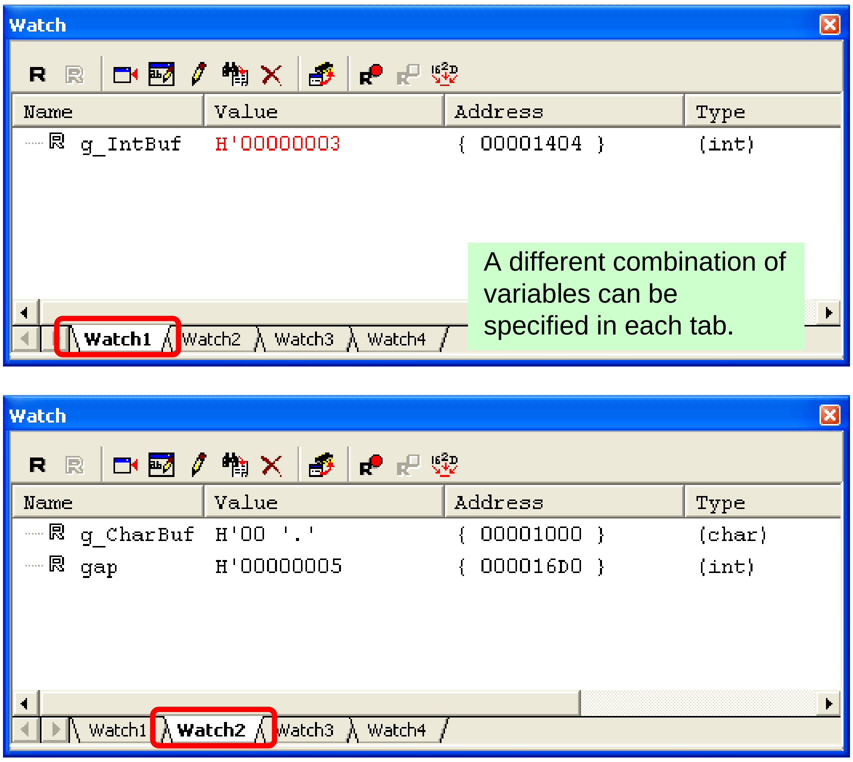


Name	Value	Address	Type
g_IntBuf	H'00000001	{ 00001404 }	(int)

Check the changes of the value.

Figure 4.26 g_IntBuf Value after Several Breaks

(7) The [Watch] window has four tabs and four combinations of variables can be monitored by switching the tabs.



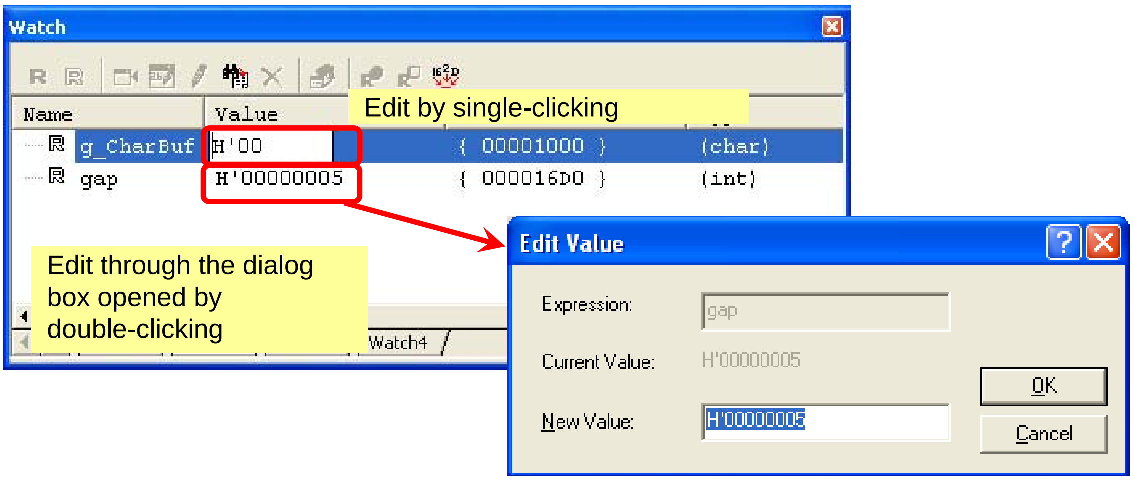
Name	Value	Address	Type
g_IntBuf	H'00000003	{ 00001404 }	(int)

A different combination of variables can be specified in each tab.

Name	Value	Address	Type
g_CharBuf	H'00 '.'	{ 00001000 }	(char)
gap	H'00000005	{ 000016D0 }	(int)

Figure 4.27 Switching Tabs in [Watch] Window

(8) Variable values can be modified in two ways as shown in Figure 4.28. One way is through in-place editing by single-clicking on the variable value. Another is through the dialog box opened by double-clicking on the variable.



Edit by single-clicking

Name	Value	Address	Type
g_CharBuf	H'00	{ 00001000 }	(char)
gap	H'00000005	{ 000016D0 }	(int)

Edit through the dialog box opened by double-clicking

Expression: gap

Current Value: H'00000005

New Value: H'00000005

Figure 4.28 Modifying Values

(9) The changes of a value during program execution can be monitored also by enabling the automatic updating function for the [Watch] window.
To try the automatic updating function, clear all breakpoints that have been specified.
After that, select a variable in the [Watch] window and then enable the automatic updating function by clicking on the [Auto Update] button as shown in Figure 4.29. For the variable for which the automatic updating function has been enabled, the R mark color changes from white to black.

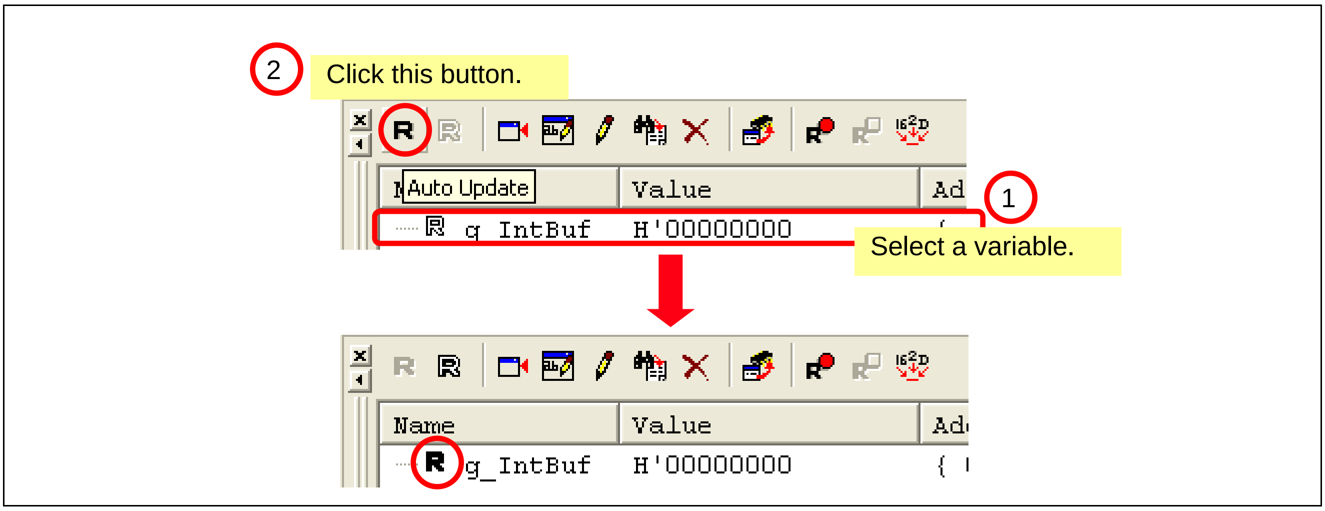
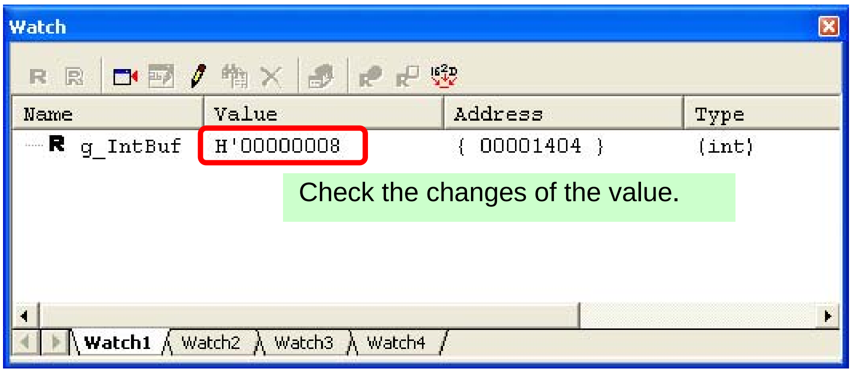


Figure 4.29 Enabling Automatic Updating for the Target Watch Item

(10)After enabling the automatic updating function, execute the program.
The changes of the g_IntBuf variable can be monitored.



Name	Value	Address	Type
g_IntBuf	H'00000008	{ 00001404 }	(int)

Figure 4.30 [Watch] Window with Automatic Updating Function Enabled

(11)The changes of values can also be recorded.
Recording of the automatically updated values can be started only while program execution is stopped. Stop program execution and then click on the record starting icon shown in Figure 4.31; recording of the automatically updated values will start.

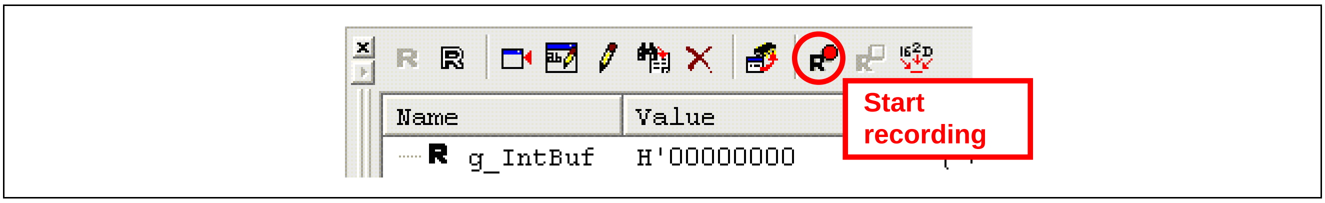
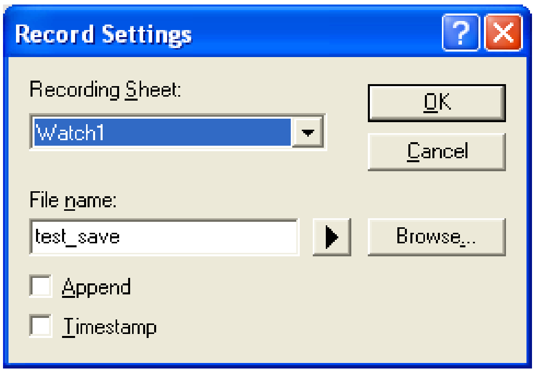


Figure 4.31 Starting Recording of Automatically Updated Values

(12)When recording is started, the [Record Settings] dialog box shown in Figure 4.32 will appear. Select the [Watch] window sheet that you want to record and enter a file name with which the record is to be saved.



Recording Sheet: Watch1

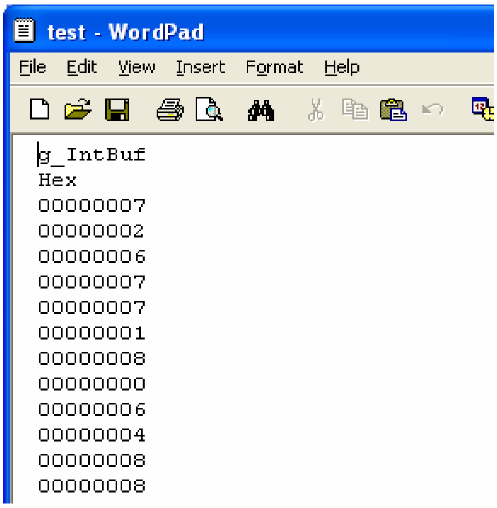
File name: test_save

□ Append

□ Timestamp

Figure 4.32 [Record Settings] Dialog Box

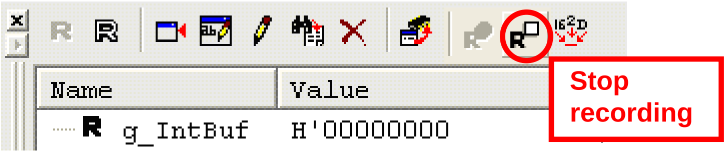
(13)After starting recording, execute the program. After checking that the value is automatically updated, click on the [Stop Recording] button with the desired timing; a text file with the file name specified in the [Record Settings] dialog box will be created in the specified folder. Figure 4.33 shows an example of the created file.



```
g_IntBuf
Hex
00000007
00000002
00000006
00000007
00000007
00000001
00000008
00000000
00000006
00000004
00000008
00000008
```

Figure 4.33 Result of Automatic Updated Value Recording

(14)To stop recording automatically updated values, click on the [Stop Recording] button.



Name	Value
g_IntBuf	H'00000000

Figure 4.34 Stopping Recording of Automatically Updated Values

4.11.2 Using the [Local] Window to Refer to or Modify Variables

Use the [Local] window to refer to local variables.

Unlike the [Watch] window, the [Local] window automatically displays the local variables used in the function that is pointed to by the PC value when program execution is stopped and allows the values to be modified.

The local variables are stored in a stack and cannot be referred to during program execution.

(1) Select [View -> Symbol -> Locals] as shown in Figure 4.35 to open the [Local] window.

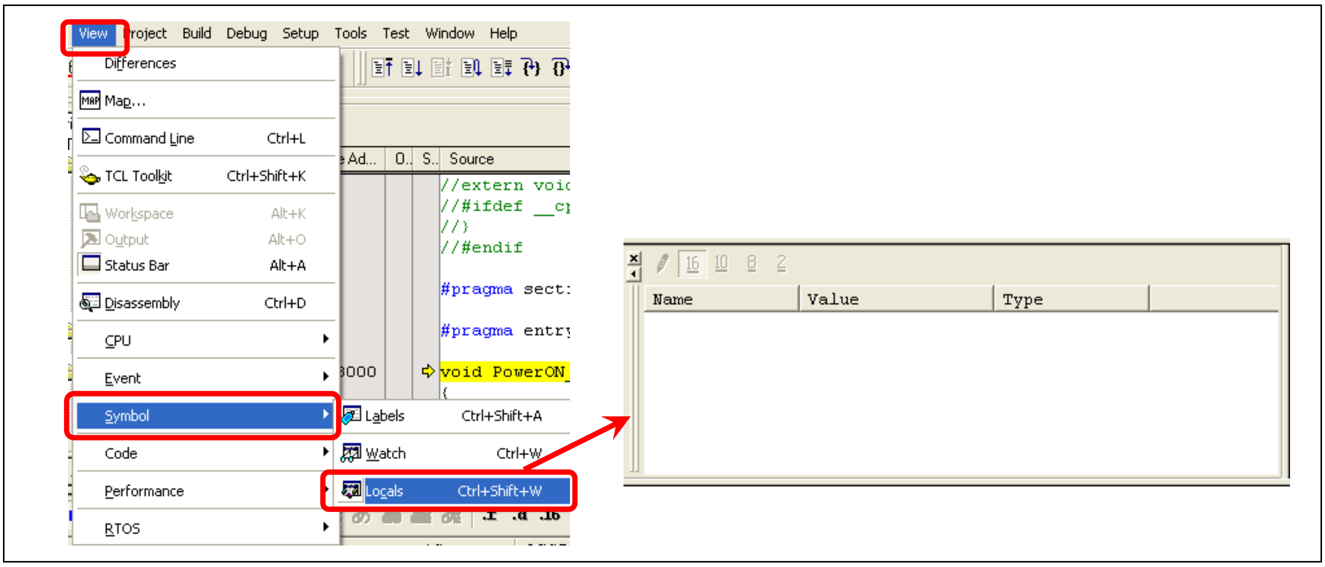
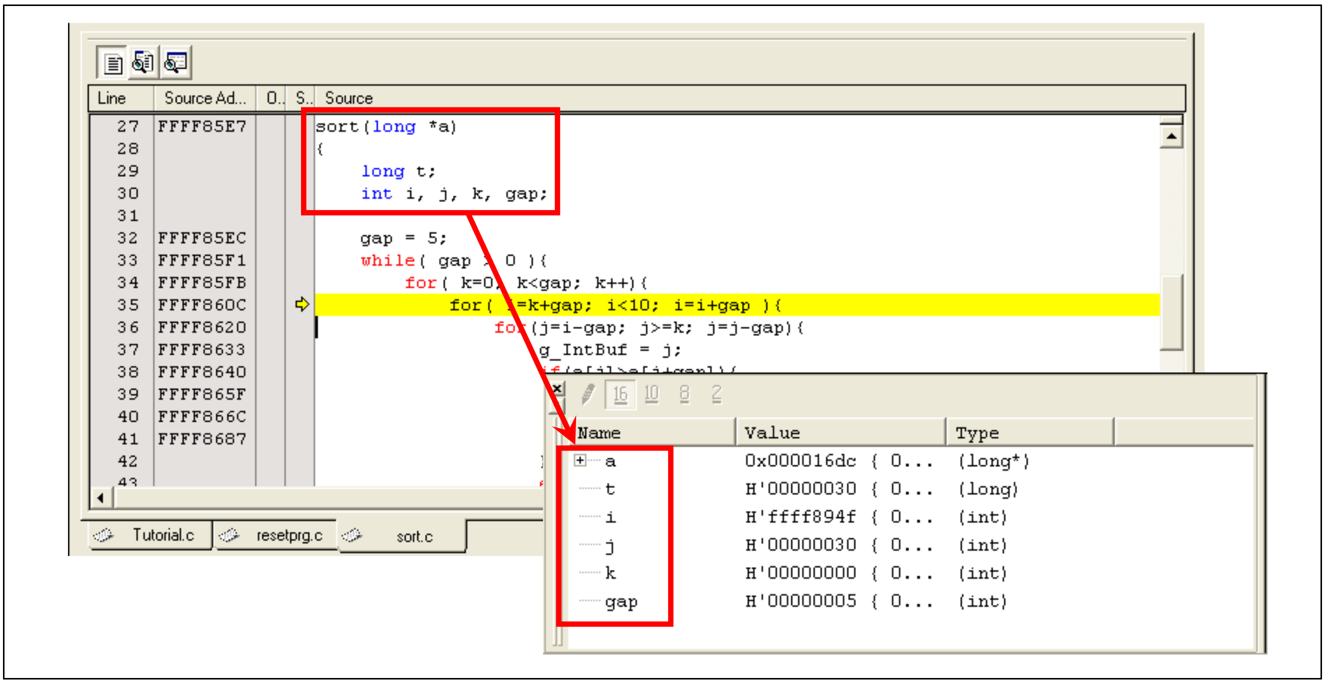


Figure 4.35 Opening the [Local] Window

(2) Execute the program and stop it with the desired timing.
The [Local] window will automatically display the variables used in the function that is pointed to by the PC value when execution is stopped as shown in Figure 4.36.



```
FFFF85E7    sort(long *a)
            {
                long t;
                int i, j, k, gap;

FFFF85EC        gap = 5;
FFFF85F1        while( gap > 0 ){
FFFF85FB            for( k=0; k<gap; k++){
FFFF860C                for( i=k+gap; i<10; i=i+gap ){
FFFF8620                    for(j=i-gap; j>=k; j=j-gap){
FFFF8633                        g_IntBuf = j;
FFFF8640
FFFF865F
FFFF866C
FFFF8687
```

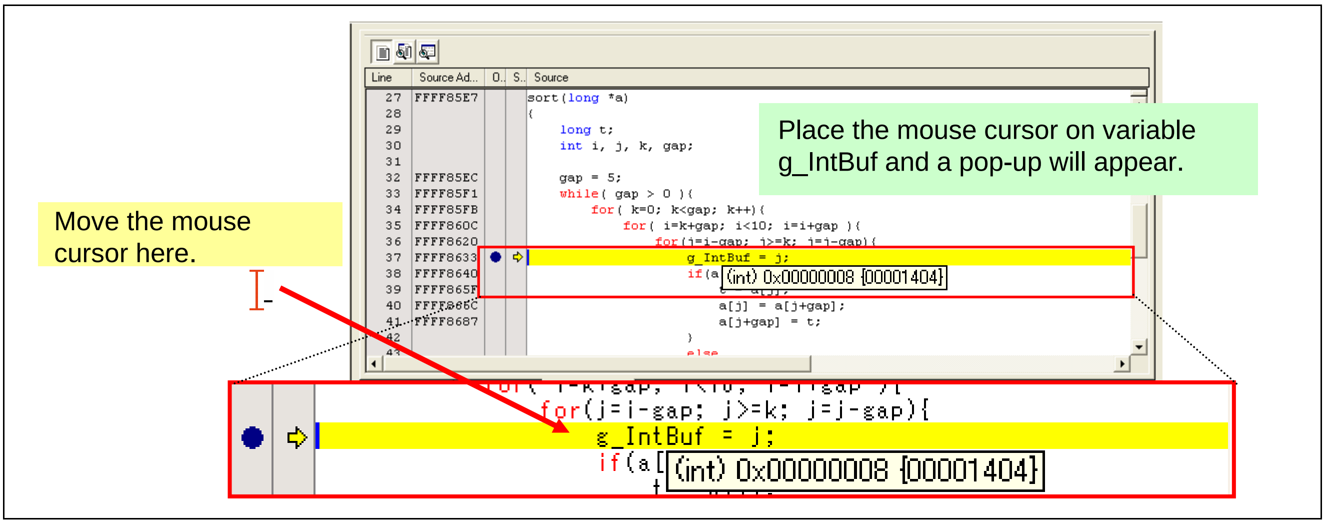
Name	Value	Type
a	0x000016dc { 0...	(long*)
t	H'00000030 { 0...	(long)
i	H'ffff894f { 0...	(int)
j	H'00000030 { 0...	(int)
k	H'00000000 { 0...	(int)
gap	H'00000005 { 0...	(int)

Figure 4.36 Referring to the [Local] Window

4.11.3 Using the Tooltip Watch Function to Refer to Variables

The tooltip watch function (watch by mouse-pointing) is available either during program execution or while execution is stopped.

Place the mouse cursor on the variable that you want to check, and the value of the variable will be displayed as shown in Figure 4.37.



```
FFFF85E7    sort(long *a)
            {
                long t;
                int i, j, k, gap;

FFFF85EC        gap = 5;
FFFF85F1        while( gap > 0 ){
FFFF85FB            for( k=0; k<gap; k++){
FFFF860C                for( i=k+gap; i<10; i=i+gap ){
FFFF8620                    for(j=i-gap; j>=k; j=j-gap){
FFFF8633                        g_IntBuf = j;
FFFF8640                        if(a (int) 0x00000008 {00001404}
FFFF865F                            t = a[j];
FFFF866C                            a[j] = a[j+gap];
FFFF8687                            a[j+gap] = t;
                                }
                                else
```

```
for(j=i-gap; j>=k; j=j-gap){
    g_IntBuf = j;
    if(a[ (int) 0x00000008 {00001404}
```

Place the mouse cursor on variable g_IntBuf and a pop-up will appear.

Move the mouse cursor here.

Figure 4.37 Tooltip Watch

4.12 Referring to or Modifying Data through the [Memory] Window

The watch function allows reference to the variables that have been declared, but use the [Memory] window to refer to or modify data in a desired address area.

(1) Select [View -> CPU -> Memory] as shown in Figure 4.38 to open the [Memory] window.

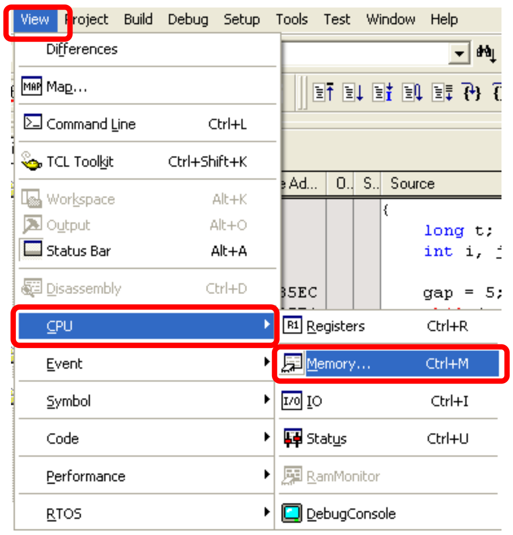
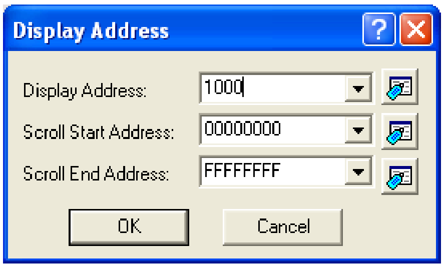


Figure 4.38 Opening the [Memory] Window

(2) The [Display Address] dialog box shown in Figure 4.39 will appear. Enter 1000H as the display start address, and the [Memory] window will open as shown in Figure 4.40.

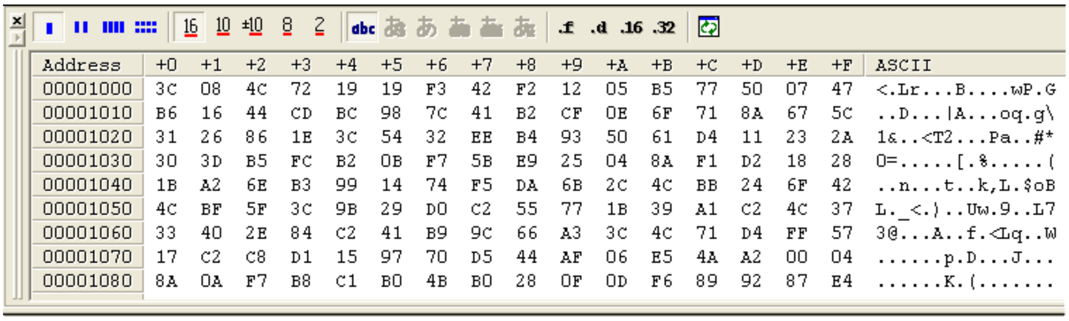

Display Address

Display Address: 1000

Scroll Start Address: 00000000

Scroll End Address: FFFFFFFF

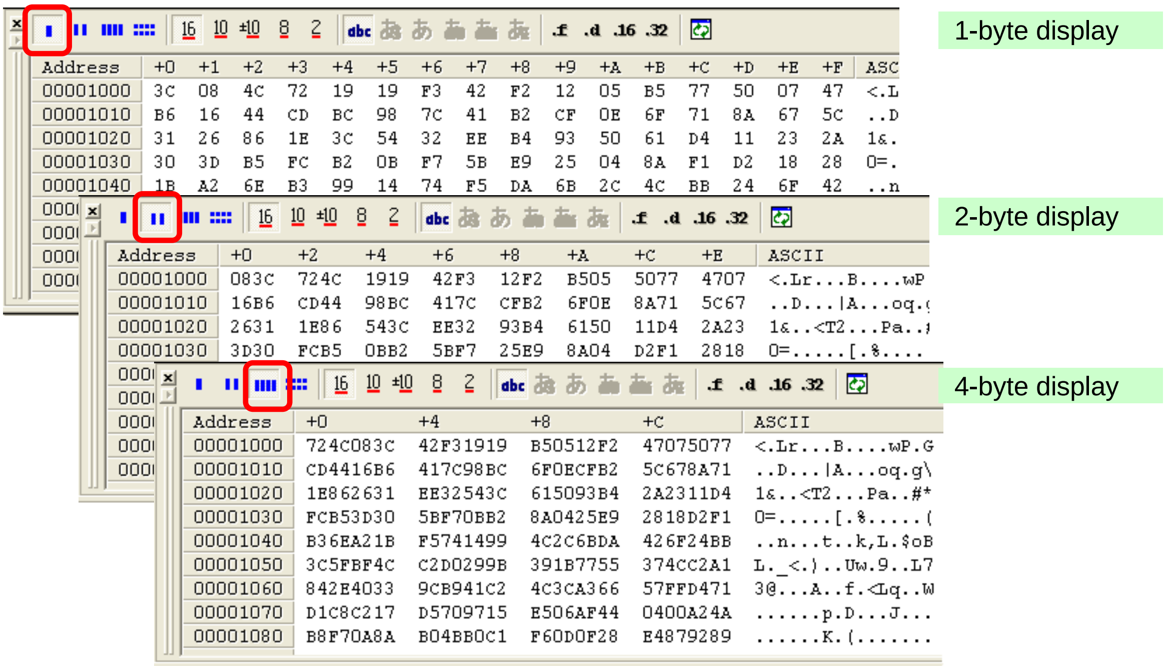
Figure 4.39 [Display Address] Dialog Box



```
Address   +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F ASCII
00001000  3C 08 4C 72 19 19 F3 42 F2 12 05 B5 77 50 07 47 <.Lr...B....wP.G
00001010  B6 16 44 CD BC 98 7C 41 B2 CF 0E 6F 71 8A 67 5C ..D...|A...oq.g\
00001020  31 26 86 1E 3C 54 32 EE B4 93 50 61 D4 11 23 2A 1&..<T2...Pa..#*
00001030  30 3D B5 FC B2 0B F7 5B E9 25 04 8A F1 D2 18 28 0=.....[.%.....(
00001040  1B A2 6E B3 99 14 74 F5 DA 6B 2C 4C BB 24 6F 42 ..n...t..k,L.$oB
00001050  4C BF 5F 3C 9B 29 D0 C2 55 77 1B 39 A1 C2 4C 37 L._<.)..Uw.9..L7
00001060  33 40 2E 84 C2 41 B9 9C 66 A3 3C 4C 71 D4 FF 57 3@...A..f.<Lq..W
00001070  17 C2 C8 D1 15 97 70 D5 44 AF 06 E5 4A A2 00 04 ......p.D...J...
00001080  8A 0A F7 B8 C1 B0 4B B0 28 0F 0D F6 89 92 87 E4 ......K.(.......
```

Figure 4.40 [Memory] Window

(3) The data display size can be changed. See Figure 4.41 for the access size change.



1-byte display

```
Address   +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F ASC
00001000  3C 08 4C 72 19 19 F3 42 F2 12 05 B5 77 50 07 47 <.L
00001010  B6 16 44 CD BC 98 7C 41 B2 CF 0E 6F 71 8A 67 5C ..D
00001020  31 26 86 1E 3C 54 32 EE B4 93 50 61 D4 11 23 2A 1&.
00001030  30 3D B5 FC B2 0B F7 5B E9 25 04 8A F1 D2 18 28 0=.
00001040  1B A2 6E B3 99 14 74 F5 DA 6B 2C 4C BB 24 6F 42 ..n
```

2-byte display

```
Address   +0   +2   +4   +6   +8   +A   +C   +E   ASCII
00001000  083C 724C 1919 42F3 12F2 B505 5077 4707 <.Lr...B....wP
00001010  16B6 CD44 98BC 417C CFB2 6F0E 8A71 5C67 ..D...|A...oq.
00001020  2631 1E86 543C EE32 93B4 6150 11D4 2A23 1&..<T2...Pa..
00001030  3D30 FCB5 0BB2 5BF7 25E9 8A04 D2F1 2818 0=.....[.%....
```

4-byte display

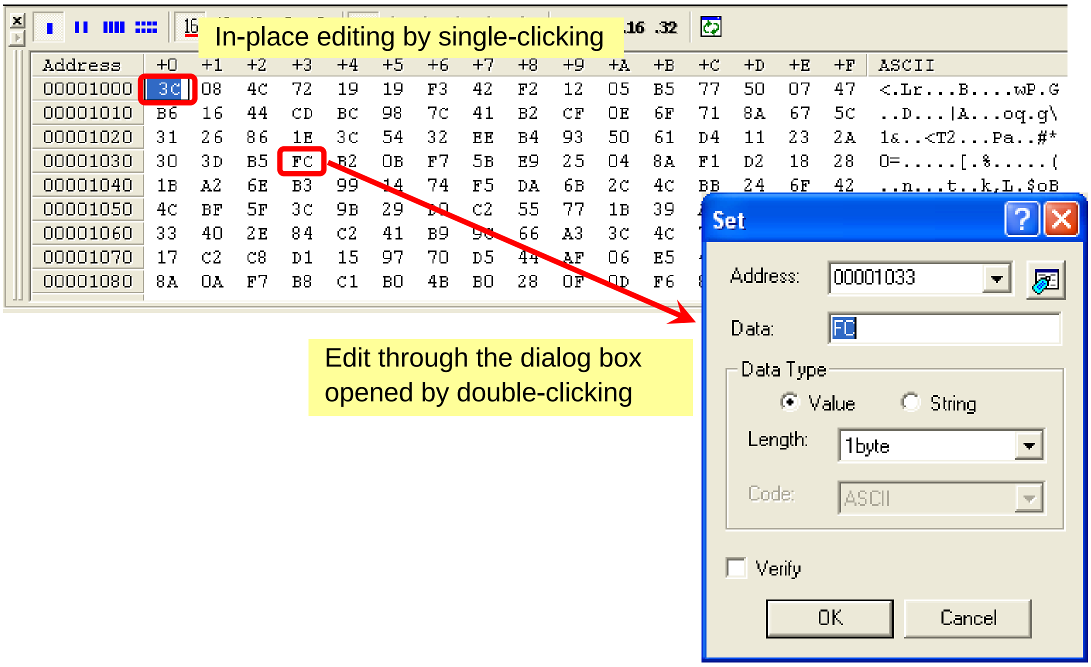
```
Address   +0       +4       +8       +C       ASCII
00001000  724C083C 42F31919 B50512F2 47075077 <.Lr...B....wP.G
00001010  CD4416B6 417C98BC 6F0ECFB2 5C678A71 ..D...|A...oq.g\
00001020  1E862631 EE32543C 615093B4 2A2311D4 1&..<T2...Pa..#*
00001030  FCB53D30 5BF70BB2 8A0425E9 2818D2F1 0=.....[.%.....(
00001040  B36EA21B F5741499 4C2C6BDA 426F24BB ..n...t..k,L.$oB
00001050  3C5FBF4C C2D0299B 391B7755 374CC2A1 L._<.)..Uw.9..L7
00001060  842E4033 9CB941C2 4C3CA366 57FFD471 3@...A..f.<Lq..W
00001070  D1C8C217 D5709715 E506AF44 0400A24A ......p.D...J...
00001080  B8F70A8A B04BB0C1 F60D0F28 E4879289 ......K.(.......
```

Figure 4.41 Changing Data Display Size in [Memory] Window

Notes: 1. In principle, data is accessed in the specified display size in the [Memory] window.

2. When the [Memory] window is used in the I/O register area, the access size allowed for the target register should always be selected as the display size. To refer to or modify the I/O register area, use the [I/O] window instead of the [Memory] window, because data is accessed in the correct access size automatically in the [I/O] window.

(4) Memory data can be modified in two ways.



In-place editing by single-clicking

```
Address   +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F ASCII
00001000  3C 08 4C 72 19 19 F3 42 F2 12 05 B5 77 50 07 47 <.Lr...B....wP.G
00001010  B6 16 44 CD BC 98 7C 41 B2 CF 0E 6F 71 8A 67 5C ..D...|A...oq.g\
00001020  31 26 86 1E 3C 54 32 EE B4 93 50 61 D4 11 23 2A 1&..<T2...Pa..#*
00001030  30 3D B5 FC B2 0B F7 5B E9 25 04 8A F1 D2 18 28 0=.....[.%.....(
00001040  1B A2 6E B3 99 14 74 F5 DA 6B 2C 4C BB 24 6F 42 ..n...t..k,L.$oB
00001050  4C BF 5F 3C 9B 29 D0 C2 55 77 1B 39
00001060  33 40 2E 84 C2 41 B9 9C 66 A3 3C 4C
00001070  17 C2 C8 D1 15 97 70 D5 44 AF 06 E5
00001080  8A 0A F7 B8 C1 B0 4B B0 28 0F 0D F6
```

Edit through the dialog box opened by double-clicking

Set

Address: 00001033

Data: FC

Data Type

● Value ○ String

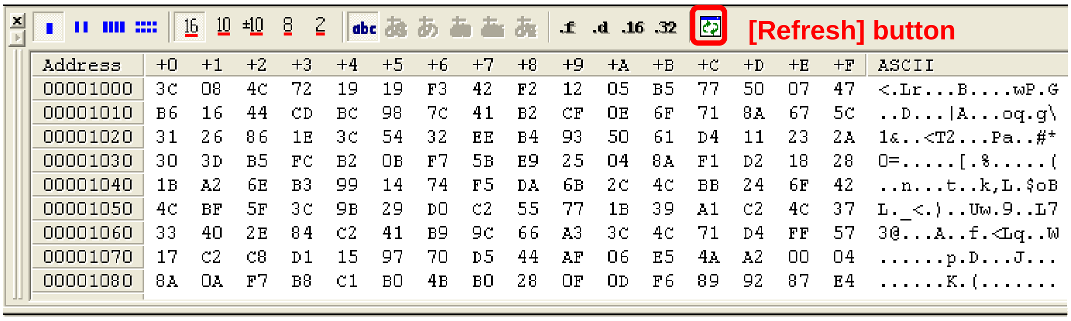
Length: 1byte

Code: ASCII

☐ Verify

Figure 4.42 Modifying Memory Data

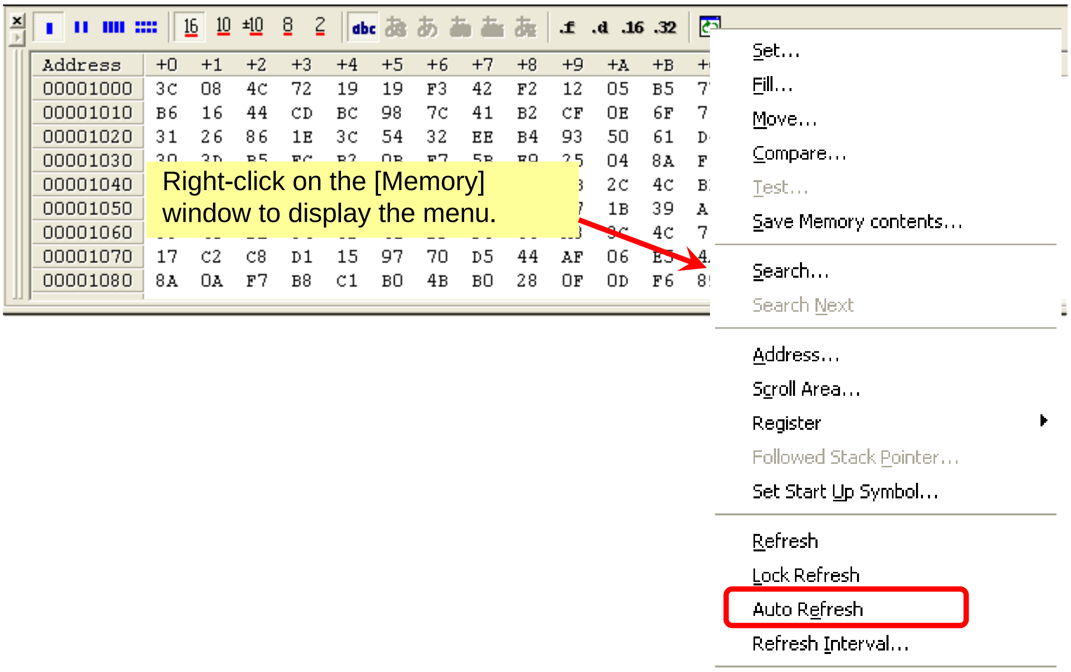
(5) The [Memory] window shows the values acquired when a break occurred. Click on the [Refresh] button to check the latest values.



```
Address   +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F ASCII
00001000  3C 08 4C 72 19 19 F3 42 F2 12 05 B5 77 50 07 47 <.Lr...B....wP.G
00001010  B6 16 44 CD BC 98 7C 41 B2 CF 0E 6F 71 8A 67 5C ..D...|A...oq.g\
00001020  31 26 86 1E 3C 54 32 EE B4 93 50 61 D4 11 23 2A 1&..<T2...Pa..#*
00001030  30 3D B5 FC B2 0B F7 5B E9 25 04 8A F1 D2 18 28 0=.....[.%.....(
00001040  1B A2 6E B3 99 14 74 F5 DA 6B 2C 4C BB 24 6F 42 ..n...t..k,L.$oB
00001050  4C BF 5F 3C 9B 29 D0 C2 55 77 1B 39 A1 C2 4C 37 L._<.)..Uw.9..L7
00001060  33 40 2E 84 C2 41 B9 9C 66 A3 3C 4C 71 D4 FF 57 3@...A..f.<Lq..W
00001070  17 C2 C8 D1 15 97 70 D5 44 AF 06 E5 4A A2 00 04 ......p.D...J...
00001080  8A 0A F7 B8 C1 B0 4B B0 28 0F 0D F6 89 92 87 E4 ......K.(.......
```

Figure 4.43 Updating the Display

(6) As with the [Watch] window, the changes of values during program execution can be checked.



```
Address   +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +
00001000  3C 08 4C 72 19 19 F3 42 F2 12 05 B5 7
00001010  B6 16 44 CD BC 98 7C 41 B2 CF 0E 6F 7
00001020  31 26 86 1E 3C 54 32 EE B4 93 50 61 D
00001030  30 3D B5 FC B2 0B F7 5B E9 25 04 8A F
00001040                                2C 4C B
00001050                                1B 39 A
00001060                                   4C 7
00001070  17 C2 C8 D1 15 97 70 D5 44 AF 06 E5 4
00001080  8A 0A F7 B8 C1 B0 4B B0 28 0F 0D F6 8
```

Right-click on the [Memory] window to display the menu.

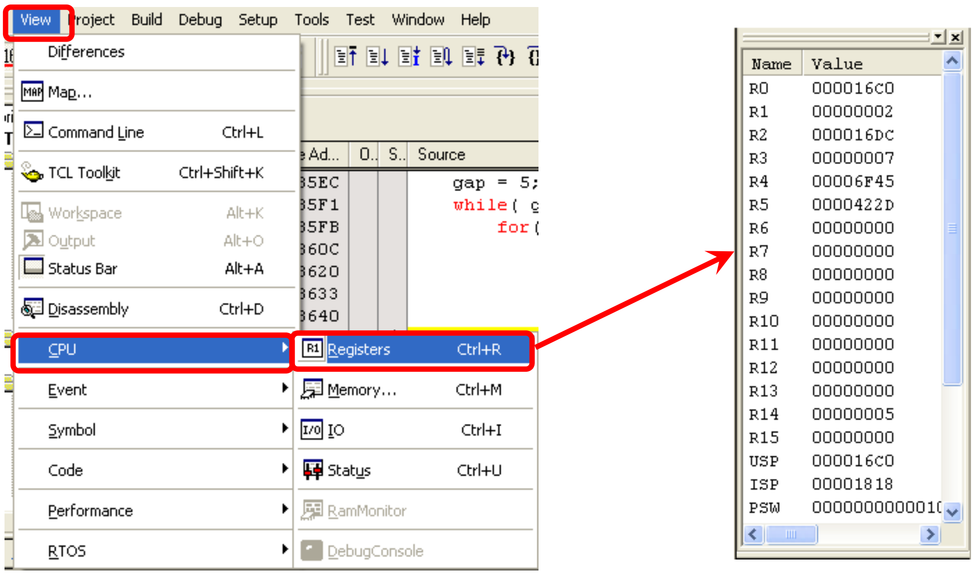
Figure 4.44 Enabling Automatic Refresh Function

4.13 Referring to or Modifying Register Values through the [Register] Window

Use the [Register] window to refer to or modify the values of general registers and control registers.

Let's try referring to the program counter (PC) value, for example.

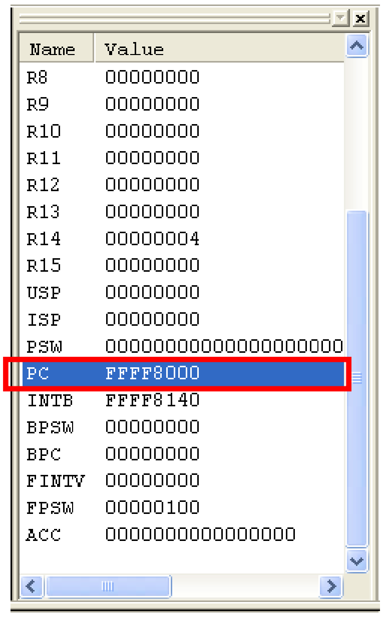
(1) Select [View -> CPU -> Registers] as shown in Figure 4.45 to open the [Register] window.



Name	Value
R0	000016C0
R1	00000002
R2	000016DC
R3	00000007
R4	00006F45
R5	0000422D
R6	00000000
R7	00000000
R8	00000000
R9	00000000
R10	00000000
R11	00000000
R12	00000000
R13	00000000
R14	00000005
R15	00000000
USP	000016C0
ISP	00001818
PSW	0000000000010

Figure 4.45 Opening the [Register] Window

(2) After the [Register] window opens, click on the [Reset] button.
As shown in Figure 4.46, the program counter is set to the value (FFFF 8000H) stored in the reset vector after a CPU reset.

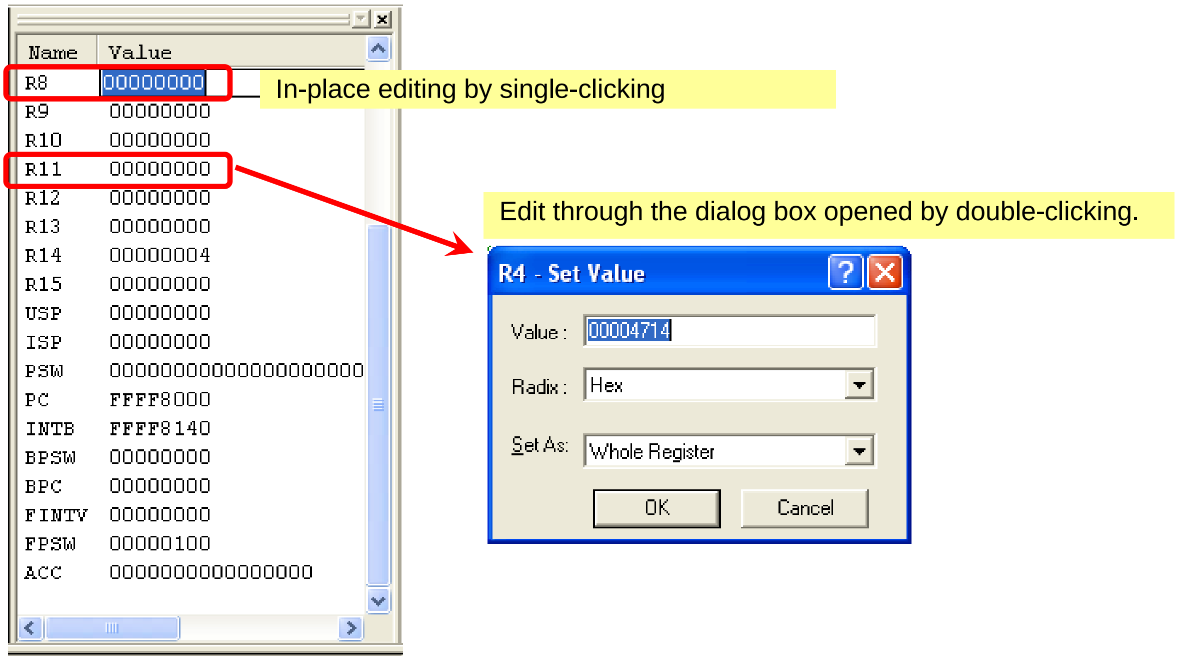


Name	Value
R8	00000000
R9	00000000
R10	00000000
R11	00000000
R12	00000000
R13	00000000
R14	00000004
R15	00000000
USP	00000000
ISP	00000000
PSW	00000000000000000000
PC	FFFF8000
INTB	FFFF8140
BPSW	00000000
BPC	00000000
FINTV	00000000
FPSW	00000100
ACC	0000000000000000

Figure 4.46 Referring to the [Register] Window

Note: The register values during execution cannot be referred to.

(3) Register values can be modified by double-clicking or single-clicking on the displayed register value.



Name	Value
R8	00000000
R9	00000000
R10	00000000
R11	00000000
R12	00000000
R13	00000000
R14	00000004
R15	00000000
USP	00000000
ISP	00000000
PSW	00000000000000000000
PC	FFFF8000
INTB	FFFF8140
BPSW	00000000
BPC	00000000
FINTV	00000000
FPSW	00000100
ACC	0000000000000000

In-place editing by single-clicking

Edit through the dialog box opened by double-clicking.

R4 - Set Value

Value : 00004714

Radix : Hex

Set As: Whole Register

Figure 4.47 Modifying Register Values

4.14 Referring to or Modifying I/O Register Values through the [I/O] Window

Use the [I/O] window to refer to or modify the values of I/O registers.

(1) Select [View -> CPU -> I/O] as shown in Figure 4.48 to open the [I/O] window.

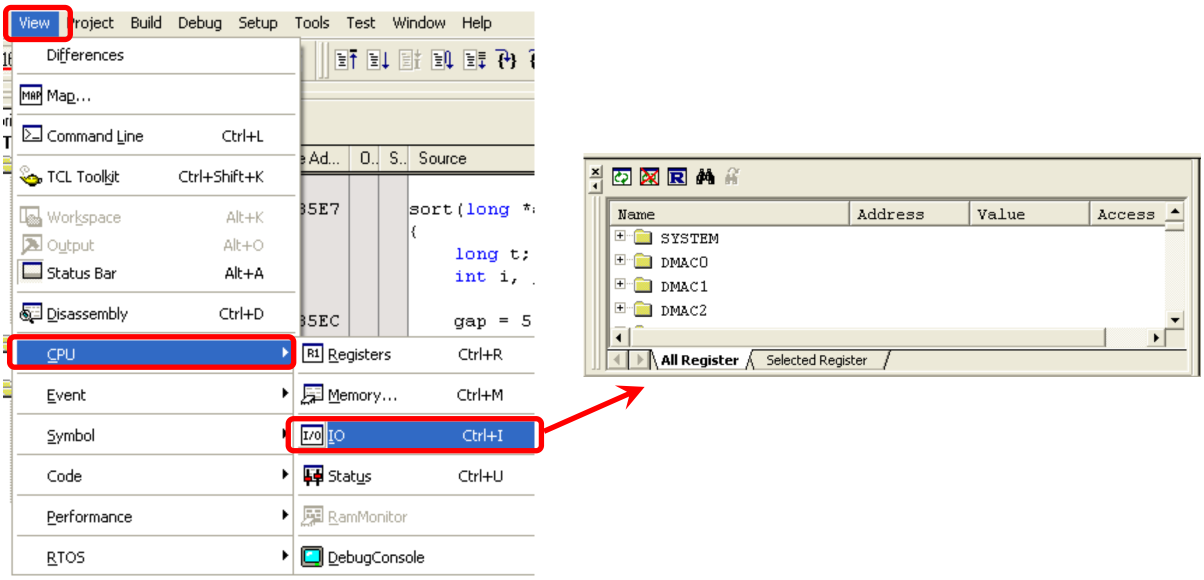
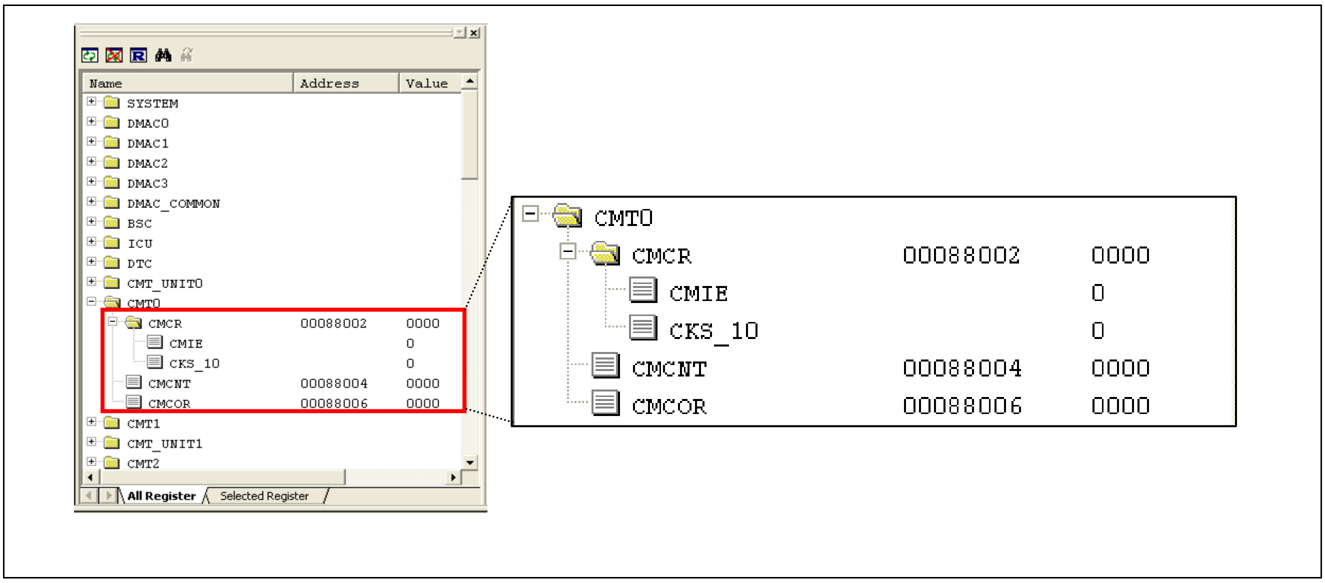


Figure 4.48 Opening the [I/O] Window

(2) Let's try referring to the compare match timer control register (CMCR) in compare match timer 0 (CMT0).

As shown in Figure 4.49, select CMT0 and expand the module display to see the register values.

This window shows the values acquired when a break occurred. To see the current value, right-click on the window to open the menu and select [Refresh] from the menu.



Name	Address	Value
SYSTEM		
DMAC0		
DMAC1		
DMAC2		
DMAC3		
DMAC_COMMON		
BSC		
ICU		
DTC		
CMT_UNIT0		
CMT0		
CMCR	00088002	0000
CMIE		0
CKS_10		0
CMCNT	00088004	0000
CMCOR	00088006	0000
CMT1		
CMT_UNIT1		
CMT2		

Figure 4.49 Referring to Register Values in the [I/O] Window

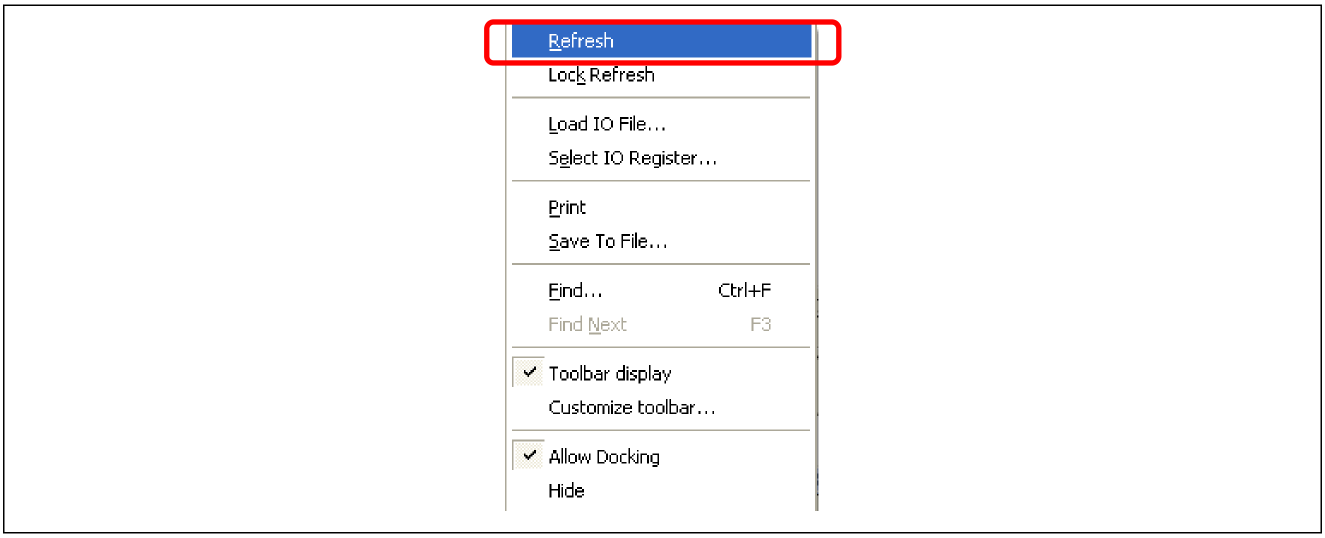
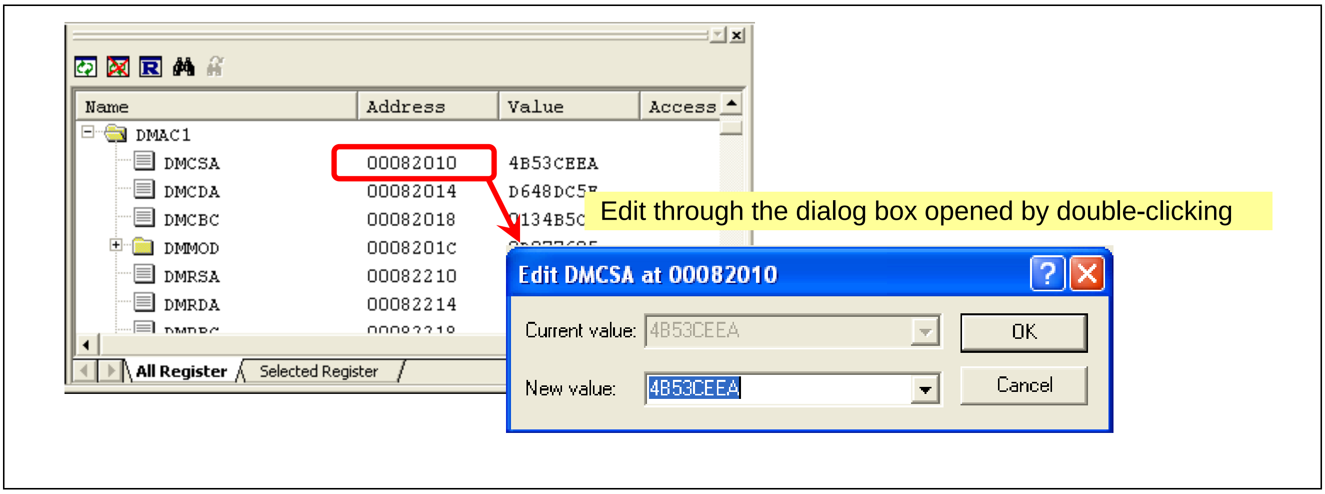


Figure 4.50 Refresh Menu

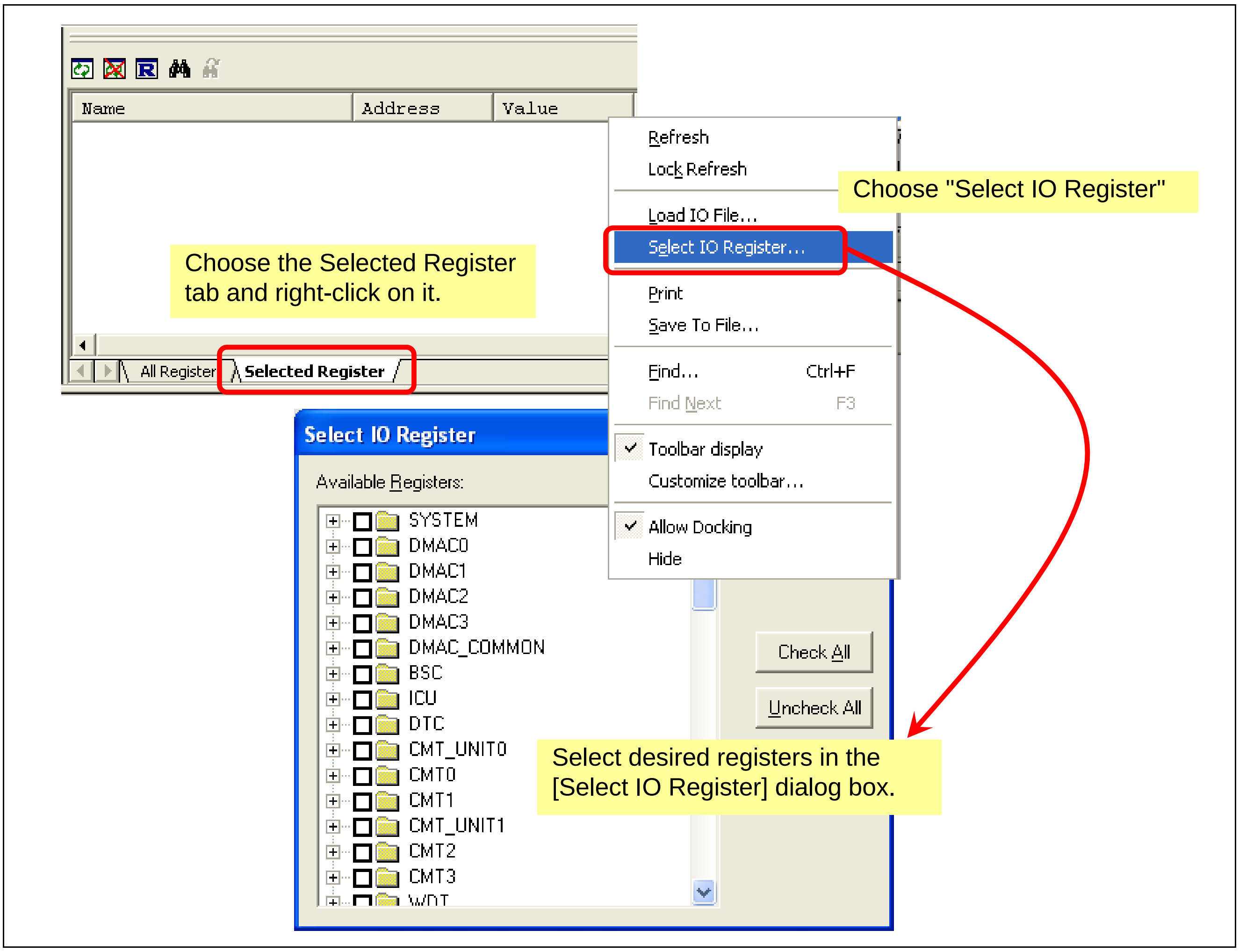
(3) I/O register values can be modified by double-clicking on the displayed I/O register value.



Edit through the dialog box opened by double-clicking

Figure 4.51 Modifying Register Values in the [I/O] Window

(4) To see only the desired registers, use the [Selected Register] tab.



Choose "Select IO Register"

Choose the Selected Register tab and right-click on it.

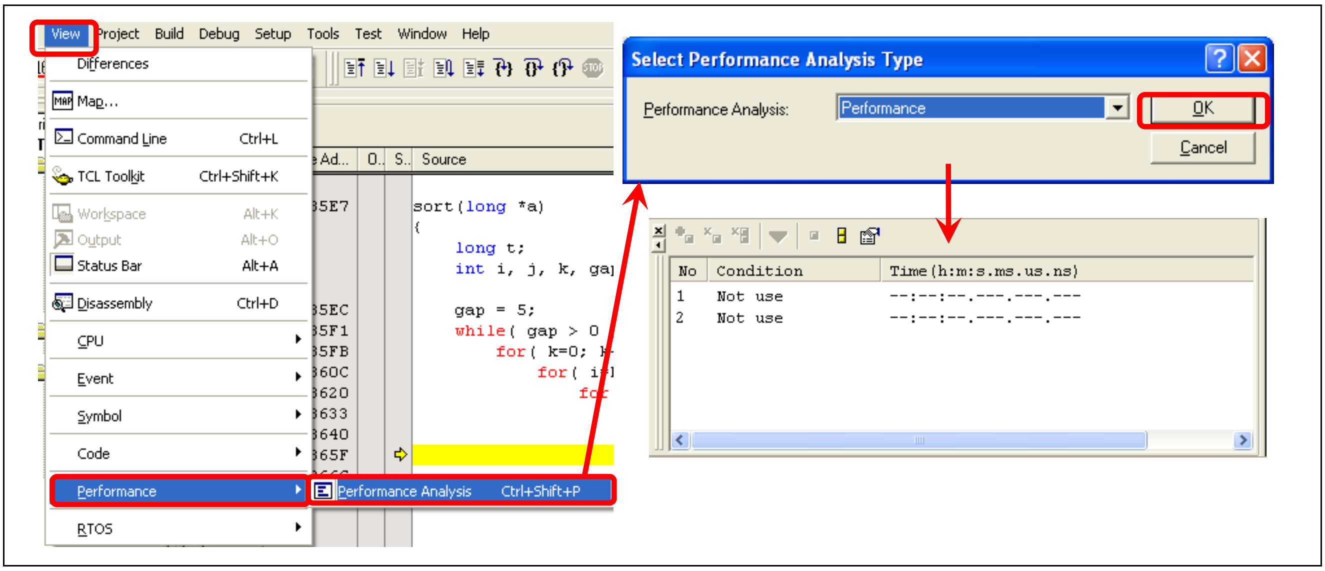
Select desired registers in the [Select IO Register] dialog box.

Figure 4.52 [Selected Register] Tab

4.15 Analyzing Performance

The E1 emulator can measure the elapsed cycles (processing time), the interrupt or exception processing cycles (processing time), or the number of accepted interrupts or exceptions in up to two user-specified sections.

(1) Open the [Performance Analysis] window.
Select [View -> Performance -> Performance Analysis] as shown in Figure 4.53 to open the [Performance Analysis] window.



```
sort(long *a)
{
    long t;
    int i, j, k, ga
    gap = 5;
    while( gap > 0
        for( k=0; k
            for( i
                fo
```

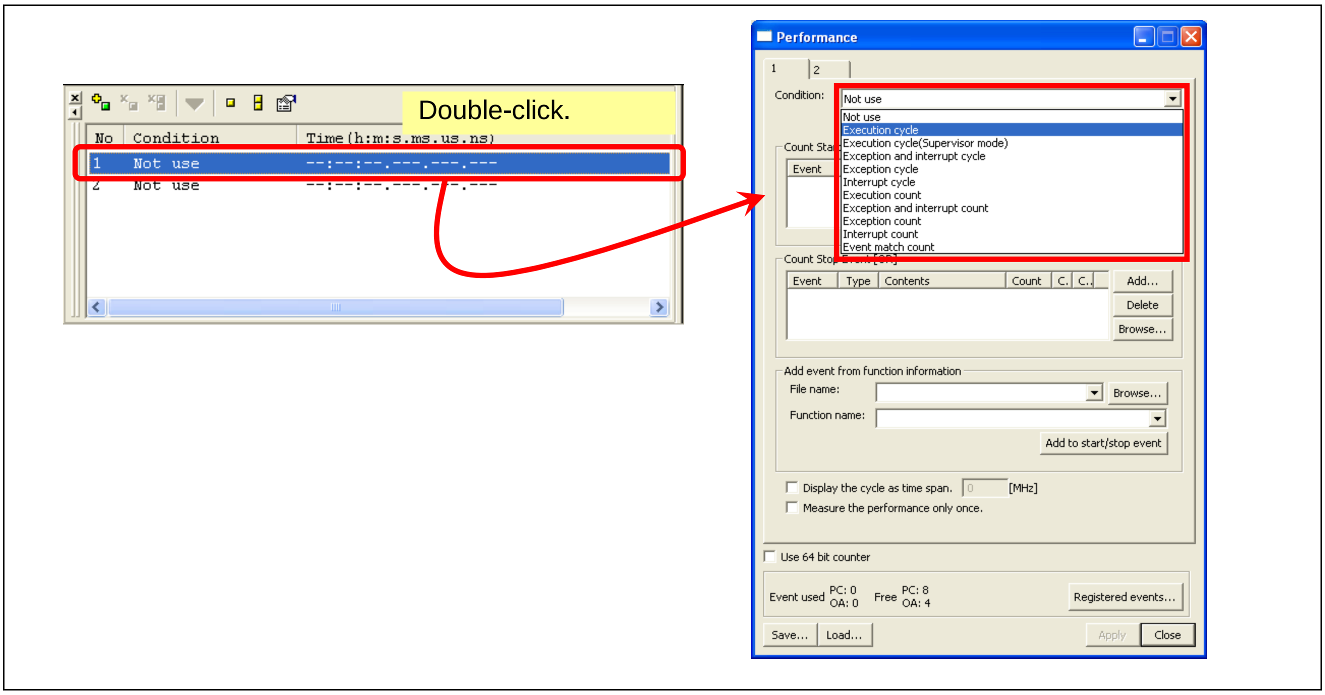
Performance Analysis: Performance

No	Condition	Time(h:m:s.ms.us.ns)
1	Not use	--:--:--.---.---.---
2	Not use	--:--:--.---.---.---

Figure 4.53 Opening the [Performance Analysis] Window

(2) As shown in Figure 4.54, select "[No.1] Not use" in the [Performance Analysis] window and double-click on it: the [Performance] dialog box will appear.

(3) The measurement items can be selected from the pull-down menu in the [Performance] dialog box.



No	Condition	Time(h:m:s.ms.us.ns)
1	Not use	--:--:--.---.---.---
2	Not use	--:--:--.---.---.---

Condition: Not use

- Not use
- Execution cycle
- Execution cycle(Supervisor mode)
- Exception and interrupt cycle
- Exception cycle
- Interrupt cycle
- Execution count
- Exception and interrupt count
- Exception count
- Interrupt count
- Event match count

☐ Display the cycle as time span. 0 [MHz]
☐ Measure the performance only once.
☐ Use 64 bit counter

Event used PC: 0 OA: 0 Free PC: 8 OA: 4

Figure 4.54 Opening the [Performance] Dialog Box

(4) Let's try measuring the execution time of the sort and change functions.

Specify the performance conditions as follows.

The count start and end events can be specified by dragging and dropping the target function names.

1 Select "Execution cycle" for the [Condition] box in the [Performance] dialog box.

2 Select "sort" in the editor window and then drag and drop it into the [Count Start Event] box in the performance condition setting dialog box.

3 Click on the [Add] button for the [Count Stop Event] box in the dialog box to open the [Event] dialog box.

4 In the event setting dialog box, enter address FFFF87E4 as the measurement end event, which is the address of line 59 in Tutorial.c.

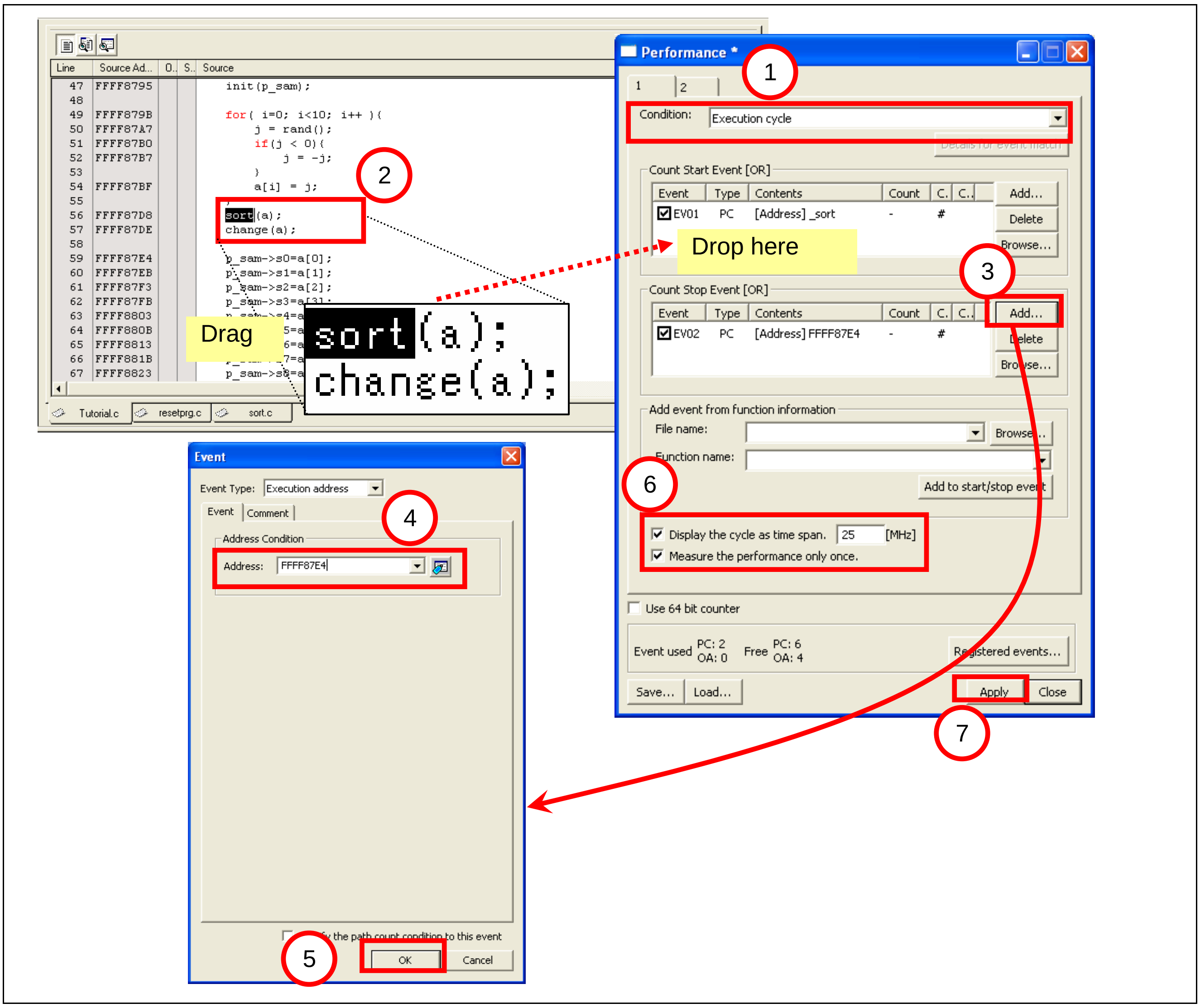
5 Click on the [OK] button in the the [Event] dialog box.

6 Choose "Display the cycle as time span" in the [Performance] dialog box, enter "25" in the frequency box, and choose "Measure the performance only once".

Choosing "Display the cycle as time span" causes the measured number of elapsed cycles to be converted into time. As the tutorial program is set up so that the CPU operates with a 25-MHz clock, enter 25 as the frequency.

Choosing "Measure the performance only once" means that only the first execution can be measured even if the specified conditions are satisfied repeatedly. When this option is not chosen, the cumulative time will be measured.

7 Check the specified conditions and click on the [Apply] button in the [Performance] dialog box to complete condition setting.



```
Line  Source Ad...  Source
  47  FFFF8795      init(p_sam);
  48
  49  FFFF879B      for( i=0; i<10; i++ ){
  50  FFFF87A7          j = rand();
  51  FFFF87B0          if(j < 0){
  52  FFFF87B7              j = -j;
  53                    }
  54  FFFF87BF          a[i] = j;
  55                }
  56  FFFF87D8      sort(a);
  57  FFFF87DE      change(a);
  58
  59  FFFF87E4      p_sam->s0=a[0];
  60  FFFF87EB      p_sam->s1=a[1];
  61  FFFF87F3      p_sam->s2=a[2];
  62  FFFF87FB      p_sam->s3=a[3];
  63  FFFF8803      p_sam->s4=a
  64  FFFF880B      p_sam->s5=a
  65  FFFF8813      p_sam->s6=a
  66  FFFF881B      p_sam->s7=a
  67  FFFF8823      p_sam->s8=a
```

```
sort(a);
change(a);
```

Performance *

Condition: Execution cycle

Count Start Event [OR]

Event	Type	Contents	Count	C.	C..
☑ EV01	PC	[Address] _sort	-	#	

Count Stop Event [OR]

Event	Type	Contents	Count	C.	C..
☑ EV02	PC	[Address] FFFF87E4	-	#	

Add event from function information

File name:

Function name:

☑ Display the cycle as time span. 25 [MHz]

☑ Measure the performance only once.

□ Use 64 bit counter

Event used PC: 2 OA: 0 Free PC: 6 OA: 4

Event

Event Type: Execution address

Address Condition

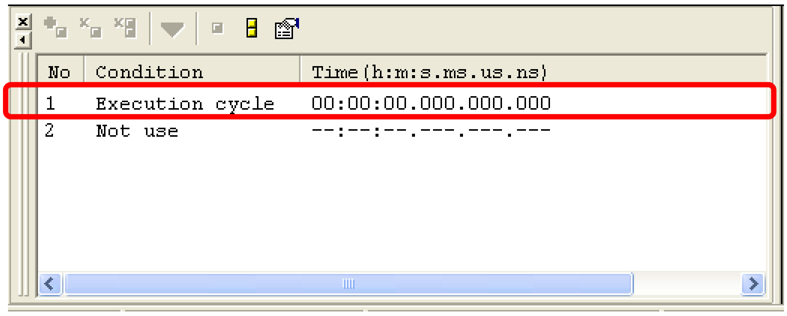
Address: FFFF87E4

□ ...fy the path count condition to this event

Figure 4.55 Specifying Performance Conditions

Note: This example assumes that the sort and change functions are called only from these addresses.

(5) Check that the settings are reflected in the [Performance Analysis] window as shown in Figure 4.56.

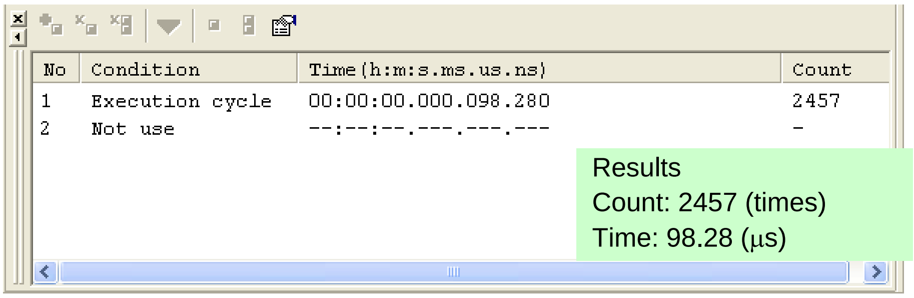


No	Condition	Time (h:m:s.ms.us.ns)
1	Execution cycle	00:00:00.000.000.000
2	Not use	--:--:--.---.---.---

Figure 4.56 [Performance Analysis] Window after Conditions are Specified

(6) After checking that the window reflects the settings, click on the [Reset Go] button to start program execution.

When the specified measurement section has been passed, the [Performance Analysis] window will be updated as shown in Figure 4.57.



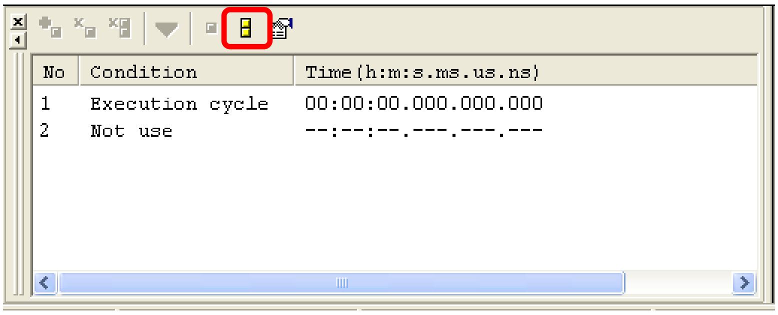
No	Condition	Time (h:m:s.ms.us.ns)	Count
1	Execution cycle	00:00:00.000.098.280	2457
2	Not use	--:--:--.---.---.---	-

Results
Count: 2457 (times)
Time: 98.28 (μs)

Figure 4.57 Performance Analysis Results

(7) To clear the measured results, wait until execution stops and then click on the [Clear All Data] button.

When the measured results are not cleared, the results of the next measurement will be accumulated on the current results.



No	Condition	Time (h:m:s.ms.us.ns)
1	Execution cycle	00:00:00.000.000.000
2	Not use	--:--:--.---.---.---

Figure 4.58 Clearing the Performance Analysis Results

4.16 Trace Function

The trace function acquires information such as PC values at branches or data access information during user program execution and stores the information in trace memory. By using the trace information, you can track the flow of application execution and examine the points where problems arise.

The E1 emulator provides the internal trace function, which uses the trace buffer in the MCU.

Execution-address and operand-access events can be used to acquire CPU-bus trace information on branches (origin and destination addresses) and data access. The acquired trace information is displayed in the [Trace] window as bus information, disassembled code, or source code.

(1) Select [View -> Code -> Trace] as shown in Figure 4.59 to open the [Trace] window.

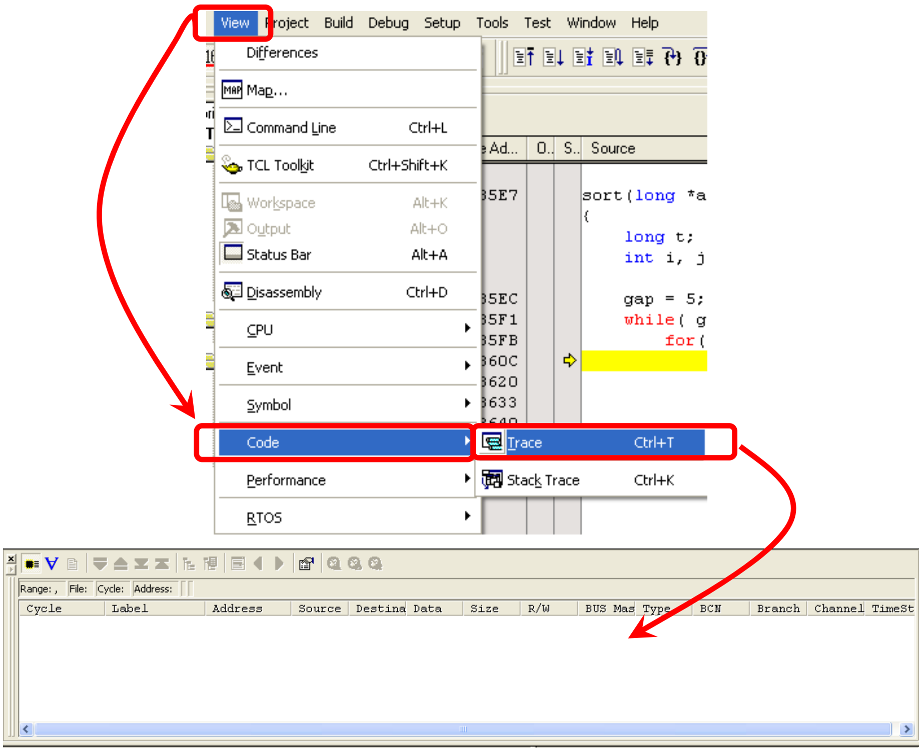


Figure 4.59 Opening the [Trace] Window

(2) Open the [Trace conditions] dialog box to specify the conditions for acquiring trace information. Click the [Acquisition...] button in the [Trace] window as shown in Figure 4.60.

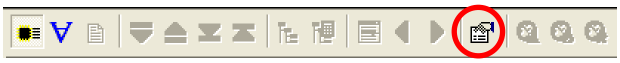


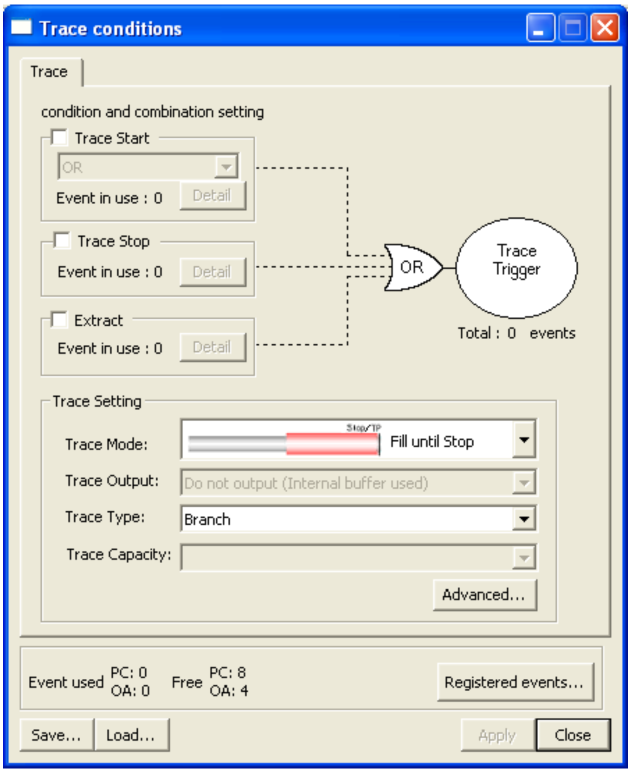
Figure 4.60 [Acquisition] Button

(3) Specify desired conditions in the [Trace conditions] dialog box.
Table 4.3 shows the conditions that can be selected in the E1 emulator.
In this example, specify conditions as shown in Figure 4.61 so that the trace function acquires the PC addresses for 256 branches immediately before the user program execution stops.

Table 4.3 Specifiable Trace Conditions (for E1 Emulator)

Trace Tab		
Trace Mode		
	Fill until stop	The acquisition of trace data continues until the program stops or an event selected to stop tracing occurs.
	Fill until full	The acquisition of trace data stops when the trace buffer becomes full.
Trace Output		
	Do not output	The trace buffer in the MCU will be used. The other options cannot be specified in the E1 emulator.
Trace Type		
	Branch	Branch information is acquired.
	Branch + Data	Branch and data-access information is acquired.
	Data	Data-access information is acquired.
Trace Capacity		
	Cannot be specified	

Note: For other trace conditions, refer to the user's manual.



Trace conditions

condition and combination setting

□ Trace Start: OR, Event in use : 0

□ Trace Stop: Event in use : 0

□ Extract: Event in use : 0

Total : 0 events

Trace Setting

Trace Mode: Fill until Stop

Trace Output: Do not output (Internal buffer used)

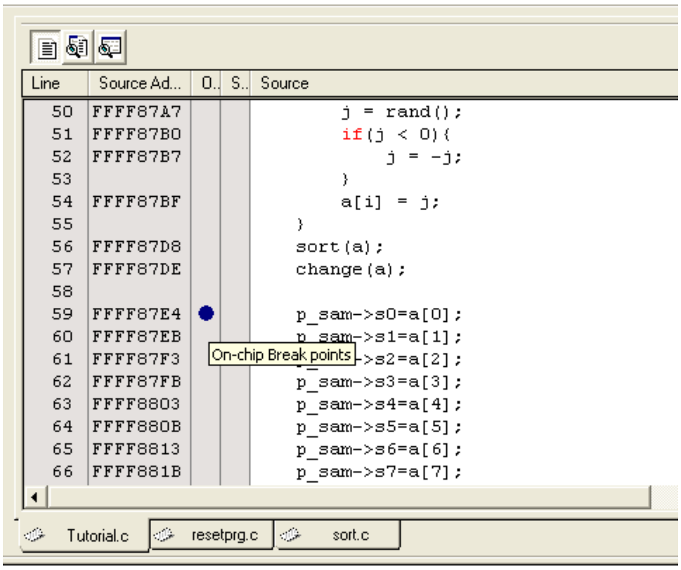
Trace Type: Branch

Trace Capacity:

Event used PC: 0 OA: 0 Free PC: 8 OA: 4

Figure 4.61 Specifying Trace Conditions

(4) To stop program execution, specify an on-chip breakpoint at line 59, which processes "p_sam->s0=a[0];", in Tutorial.c as shown in Figure 4.62. For the breakpoint setting procedure, refer to section 4.10, Break Functions.

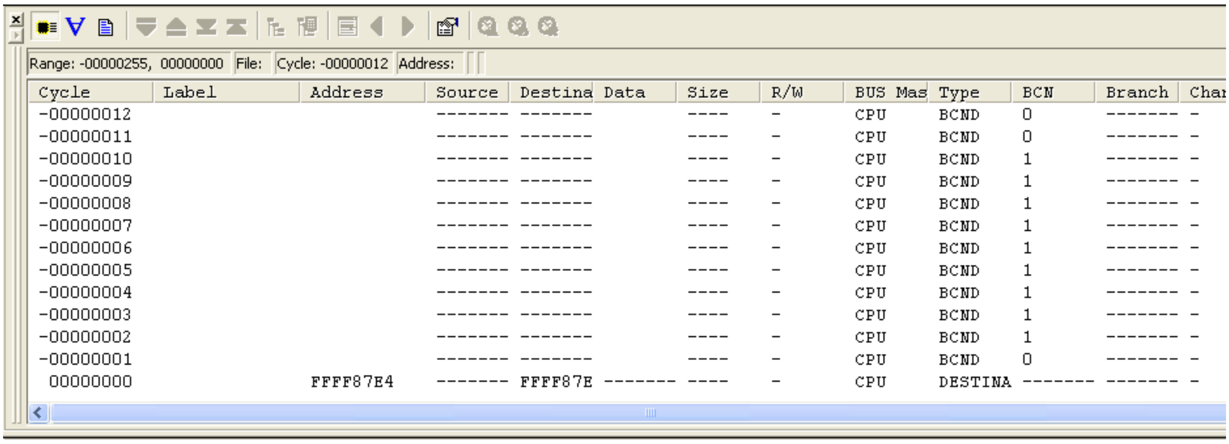


```
Line  Source Ad...  O.. S..  Source
 50   FFFF87A7                       j = rand();
 51   FFFF87B0                       if(j < 0){
 52   FFFF87B7                           j = -j;
 53                                  }
 54   FFFF87BF                       a[i] = j;
 55                              }
 56   FFFF87D8                   sort(a);
 57   FFFF87DE                   change(a);
 58
 59   FFFF87E4      ●            p_sam->s0=a[0];
 60   FFFF87EB                   p_sam->s1=a[1];
 61   FFFF87F3                   p_sam->s2=a[2];
 62   FFFF87FB                   p_sam->s3=a[3];
 63   FFFF8803                   p_sam->s4=a[4];
 64   FFFF880B                   p_sam->s5=a[5];
 65   FFFF8813                   p_sam->s6=a[6];
 66   FFFF881B                   p_sam->s7=a[7];
```

Figure 4.62 Setting a Breakpoint

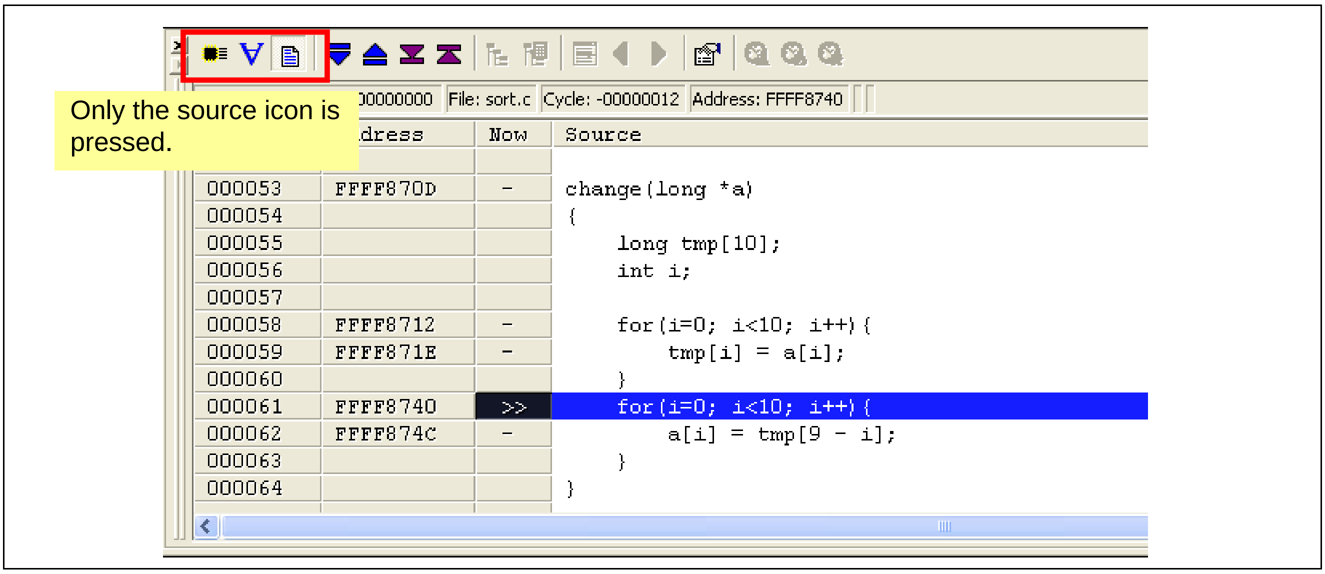
(5) Click on the [Reset Go] button to start program execution.
When execution is temporarily stopped at the specified breakpoint, the [Trace] window will be updated. The [Trace] window shows the track of program execution.

(6) Click on the display mode switching button to change from the bus display mode (Figure 4.63) to the source display mode (Figure 4.64).



```
Range: -00000255, 00000000  File:  Cycle: -00000012  Address:

Cycle      Label    Address   Source  Destina Data    Size  R/W  BUS Mas Type    BCN     Branch  Char
-00000012                     ------- -------         ----  -    CPU     BCND    0       ------- -
-00000011                     ------- -------         ----  -    CPU     BCND    0       ------- -
-00000010                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000009                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000008                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000007                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000006                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000005                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000004                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000003                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000002                     ------- -------         ----  -    CPU     BCND    1       ------- -
-00000001                     ------- -------         ----  -    CPU     BCND    0       ------- -
 00000000           FFFF87E4  ------- FFFF87E ------- ----  -    CPU     DESTINA ------- ------- -
```

Figure 4.63 [Trace] Window after a Break (Bus Display Mode)



Only the source icon is pressed.

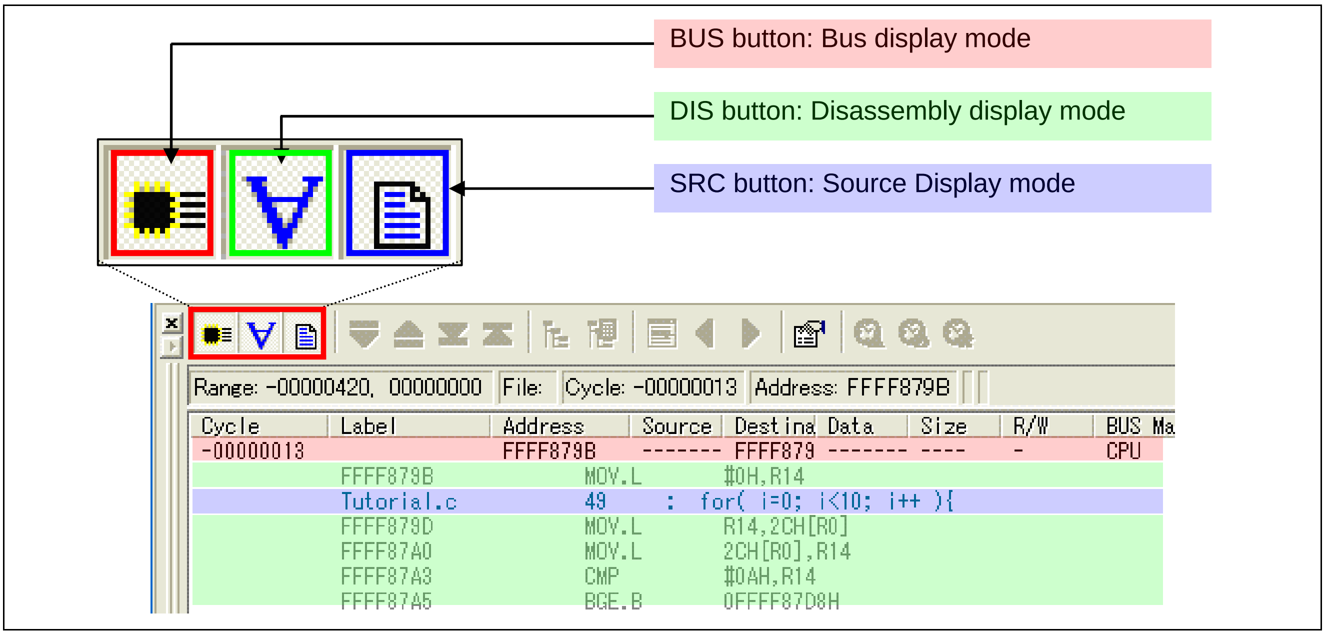
```
)0000000  File: sort.c  Cycle: -00000012  Address: FFFF8740
.dress            Now   Source

000053  FFFF870D   -    change(long *a)
000054                  {
000055                      long tmp[10];
000056                      int i;
000057
000058  FFFF8712   -        for(i=0; i<10; i++){
000059  FFFF871E   -            tmp[i] = a[i];
000060                      }
000061  FFFF8740   >>       for(i=0; i<10; i++){
000062  FFFF874C   -            a[i] = tmp[9 - i];
000063                      }
000064                  }
```

Figure 4.64 Source Display Mode

(7) The [Trace] window display modes are shown in Figure 4.65.

Figure 4.65 also shows a window display example where the bus display, disassembly display, and source display modes are all enabled.

Use these trace display mode switching buttons to select the information displayed in the [Trace] window.



BUS button: Bus display mode

DIS button: Disassembly display mode

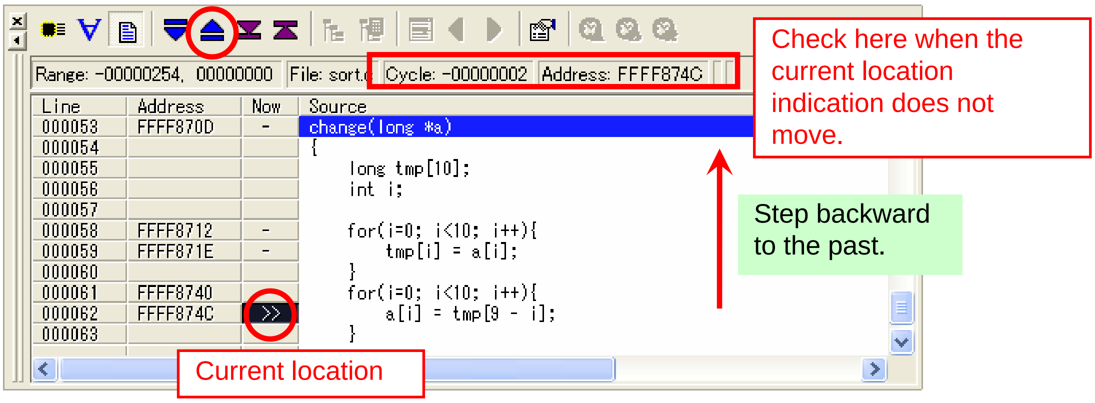
SRC button: Source Display mode

```
Range: -00000420, 00000000  File:  Cycle: -00000013  Address: FFFF879B
Cycle      Label       Address    Source   Destina Data     Size  R/W  BUS Ma
-00000013              FFFF879B   -------  FFFF879 -------  ----  -    CPU
           FFFF879B          MOV.L       #0H,R14
           Tutorial.c        49      :  for( i=0; i<10; i++ ){
           FFFF879D          MOV.L       R14,2CH[R0]
           FFFF87A0          MOV.L       2CH[R0],R14
           FFFF87A3          CMP         #0AH,R14
           FFFF87A5          BGE.B       0FFFF87D8H
```

Figure 4.65 [Trace] Window Display Modes

(8) In the source display mode, you can use the [Step Backward] button, which analyzes the trace information and tracks the program execution history in reverse as shown in Figure 4.66, and the [Step Forward] button, which tracks forward as shown in Figure 4.67.

In this tutorial example, the current location may be in the loop in the "change" function and in some cases, the indication of the current location will not move even when a step button is clicked. In this case, check the Cycle and Address in the top right of the window. If these values change when a step button is clicked, the current location is in the loop.



Range: -00000254, 00000000 | File: sort.c | Cycle: -00000002 | Address: FFFF874C

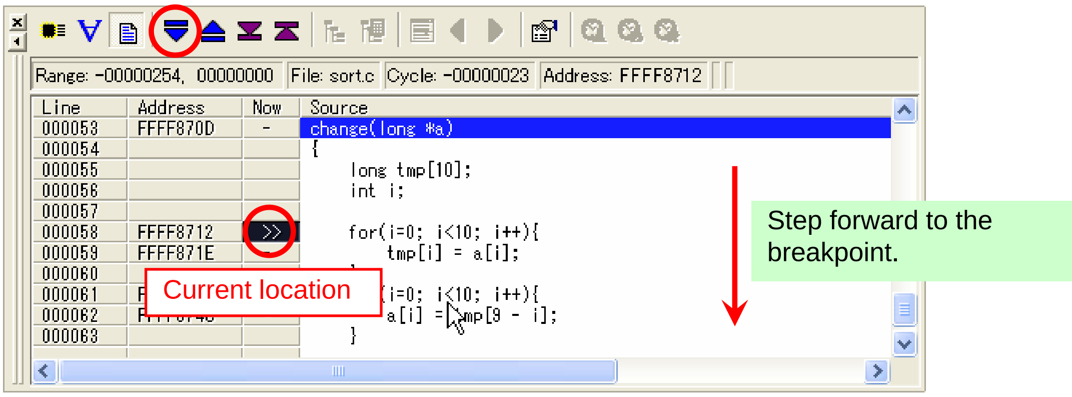
Line	Address	Now	Source
000053	FFFF870D	-	change(long *a)
000054			{
000055			long tmp[10];
000056			int i;
000057			
000058	FFFF8712	-	for(i=0; i<10; i++){
000059	FFFF871E	-	tmp[i] = a[i];
000060			}
000061	FFFF8740		for(i=0; i<10; i++){
000062	FFFF874C	>>	a[i] = tmp[9 - i];
000063			}

Check here when the current location indication does not move.

Step backward to the past.

Current location

Figure 4.66 Step Backward



Range: -00000254, 00000000 | File: sort.c | Cycle: -00000023 | Address: FFFF8712

Line	Address	Now	Source
000053	FFFF870D	-	change(long *a)
000054			{
000055			long tmp[10];
000056			int i;
000057			
000058	FFFF8712	>>	for(i=0; i<10; i++){
000059	FFFF871E		tmp[i] = a[i];
000060			
000061	F		(i=0; i<10; i++){
000062	F		a[i] = tmp[9 - i];
000063			}

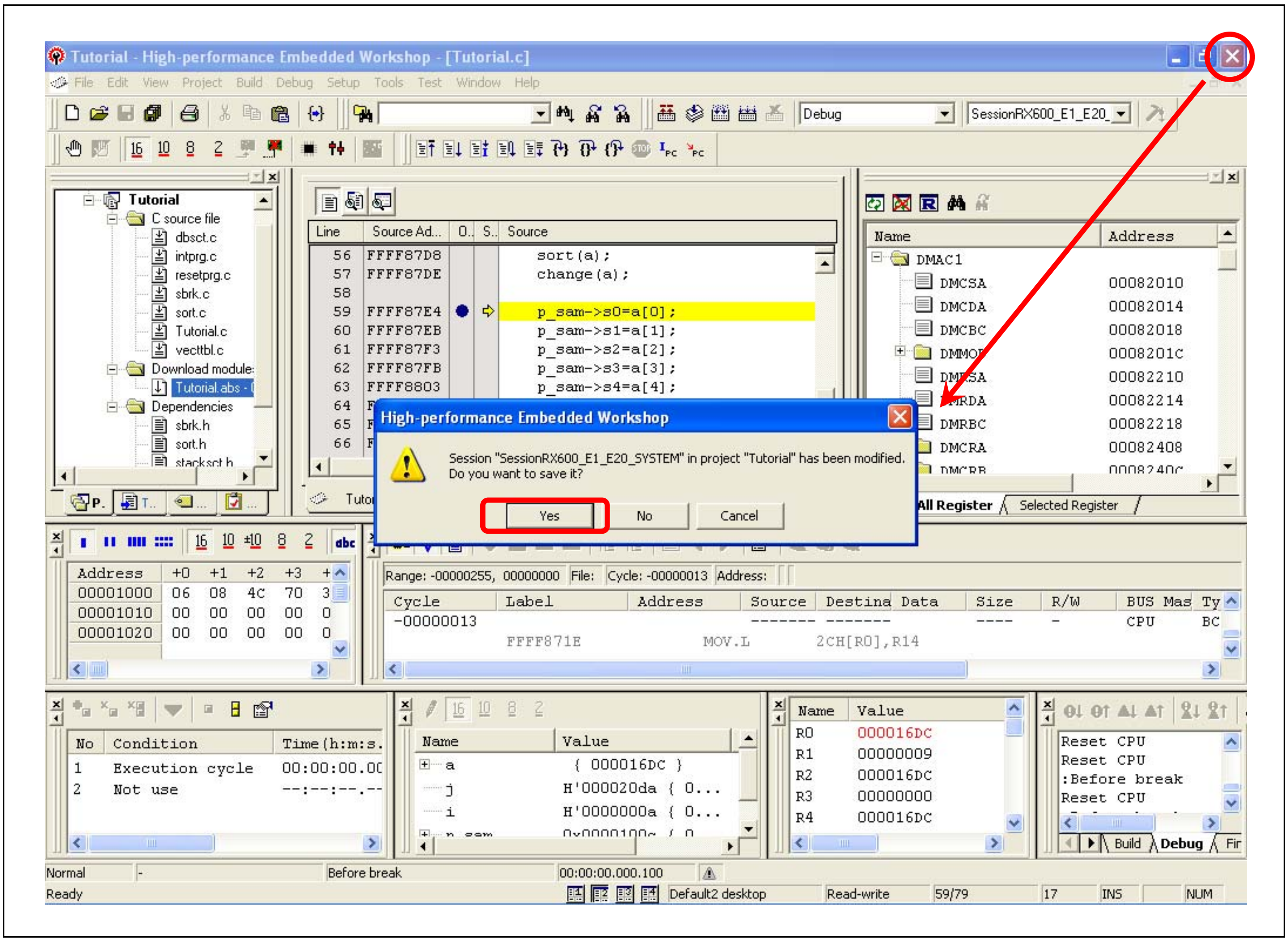
Step forward to the breakpoint.

Current location

Figure 4.67 Step Forward

4.17 Terminating Debugging

To terminate debugging, click on the × mark in the top right of the window and a dialog box will open. Click the [Yes] button to save the window status; you can start debugging with the saved window status at the next time.



High-performance Embedded Workshop

Session "SessionRX600_E1_E20_SYSTEM" in project "Tutorial" has been modified.
Do you want to save it?

Yes No Cancel

```
Line  Source Ad...  O..  S..  Source
 56   FFFF87D8                    sort(a);
 57   FFFF87DE                    change(a);
 58
 59   FFFF87E4      ●    ⇨        p_sam->s0=a[0];
 60   FFFF87EB                    p_sam->s1=a[1];
 61   FFFF87F3                    p_sam->s2=a[2];
 62   FFFF87FB                    p_sam->s3=a[3];
 63   FFFF8803                    p_sam->s4=a[4];
```

```
Name        Address
DMAC1
  DMCSA     00082010
  DMCDA     00082014
  DMCBC     00082018
  DMMOD     0008201C
  DMRSA     00082210
  DMRDA     00082214
  DMRBC     00082218
  DMCRA     00082408
  DMCRB     0008240C
```

```
Address   +0 +1 +2 +3
00001000  06 08 4C 70 3
00001010  00 00 00 00 0
00001020  00 00 00 00 0
```

```
Range: -00000255, 00000000  File:  Cycle: -00000013  Address:
Cycle      Label   Address   Source   Destina  Data   Size  R/W  BUS Mas  Ty
-00000013                    -------  -------         ----  -    CPU      BC
           FFFF871E          MOV.L      2CH[R0],R14
```

```
No  Condition        Time(h:m:s.
1   Execution cycle  00:00:00.00
2   Not use          --:--:--.--
```

```
Name       Value
+ a        { 000016DC }
  j        H'000020da { 0...
  i        H'0000000a { 0...
+ p_sam    0x0000100c { 0
```

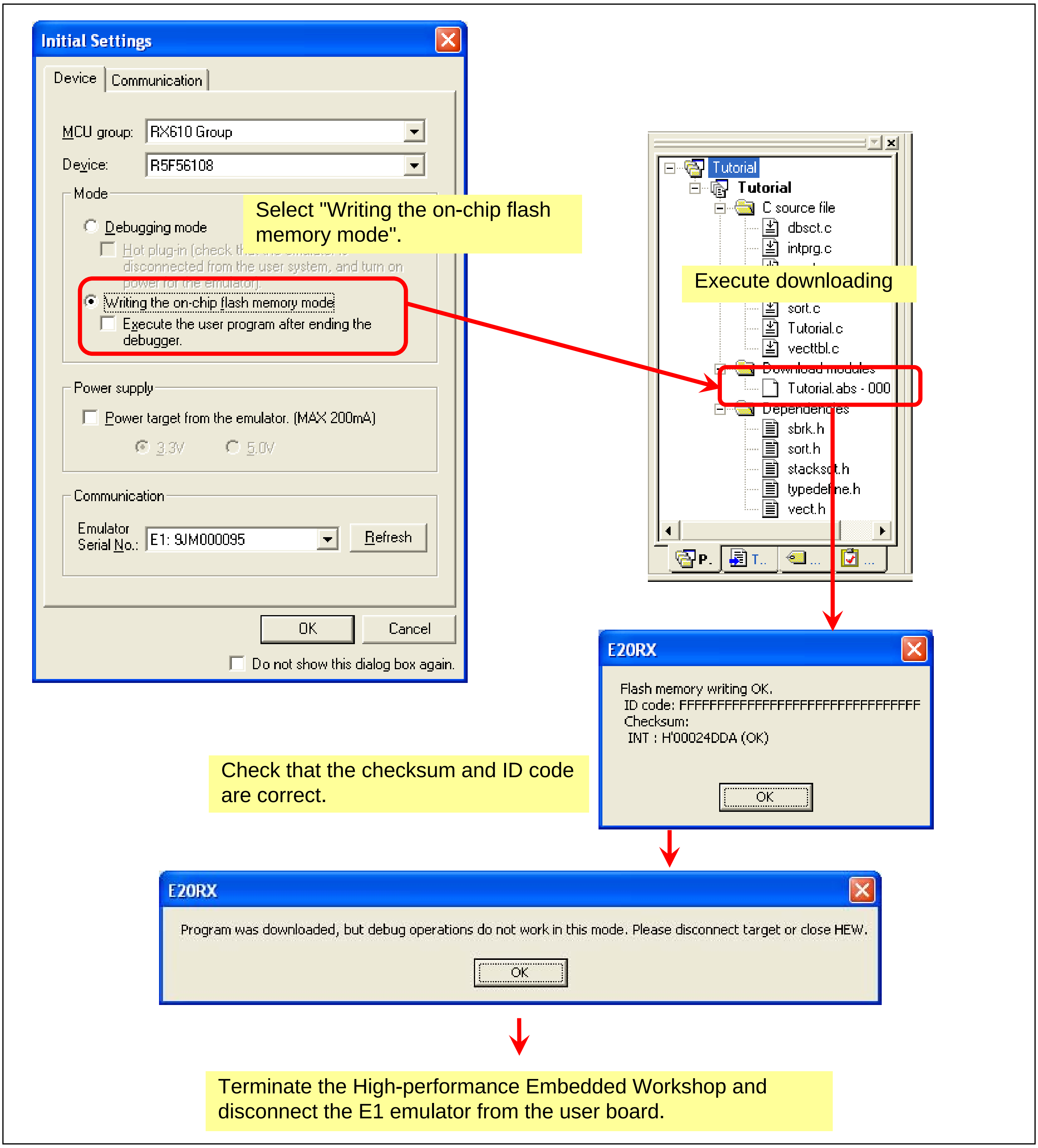
```
Name  Value
R0    000016DC
R1    00000009
R2    000016DC
R3    00000000
R4    000016DC
```

```
Reset CPU
Reset CPU
:Before break
Reset CPU
```

Figure 4.68 Terminating Debugging

4.18 Operating the User Board as a Stand-Alone Unit

When operating the user board as a stand-alone unit after completing debugging, download the program by using the flash-memory programmer mode as shown in the following figure (the steps described in the above sections are omitted).



Initial Settings

Device | Communication

MCU group: RX610 Group

Device: R5F56108

Mode

○ Debugging mode

□ Hot plug-in (check th... disconnected from the user system, and turn on power for the emulator).

● Writing the on-chip flash memory mode

□ Execute the user program after ending the debugger.

Power supply

□ Power target from the emulator. (MAX 200mA)

● 3.3V ○ 5.0V

Communication

Emulator Serial No.: E1: 9JM000095

□ Do not show this dialog box again.

Select "Writing the on-chip flash memory mode".

Execute downloading

E20RX

```
Flash memory writing OK.
 ID code: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
 Checksum:
  INT : H'00024DDA (OK)
```

Check that the checksum and ID code are correct.

E20RX

Program was downloaded, but debug operations do not work in this mode. Please disconnect target or close HEW.

Terminate the High-performance Embedded Workshop and disconnect the E1 emulator from the user board.

Figure 4.69 Procedure for Operating the User Board as a Stand-Alone Unit

5. Supplementary Information

5.1 Emulator-Related Documents

The E1 emulator and High-performance Embedded Workshop also have a variety of useful features other than those introduced in this guide. For details on the specifications of these products, refer to the documents given below.

(1) High-performance Embedded Workshop User's Manual
This document describes the functions of the High-performance Embedded Workshop (building, file handling, common debugging functions, and so on).

(2) RX Family E1/E20 Emulator User's Manual
This document describes procedures and usage notes that are common to the E1 and E20 emulators.
For debugging functions specific to the E1 or E20, refer to E1/E20 Emulator User's Manual: Additional Document for User's Manual.

(3) E1/E20 Emulator User's Manual: Additional Document for User's Manual (Supplementary Information on Using the RX610 Group)
This document contains notes on usage of the E1 or E20 emulator in combination with RX610-group MCUs. A separate document is provided for each group of MCUs.

(4) RX Family C/C++ Compiler Package User's Manual
This document contains descriptions of the C/C++ compiler, assembler, and linkage editor.

To refer to these documents, open the [Start] menu and select [All Programs -> Renesas -> High-performance Embedded Workshop -> Manual Navigator]. This invokes the Manual Navigator.

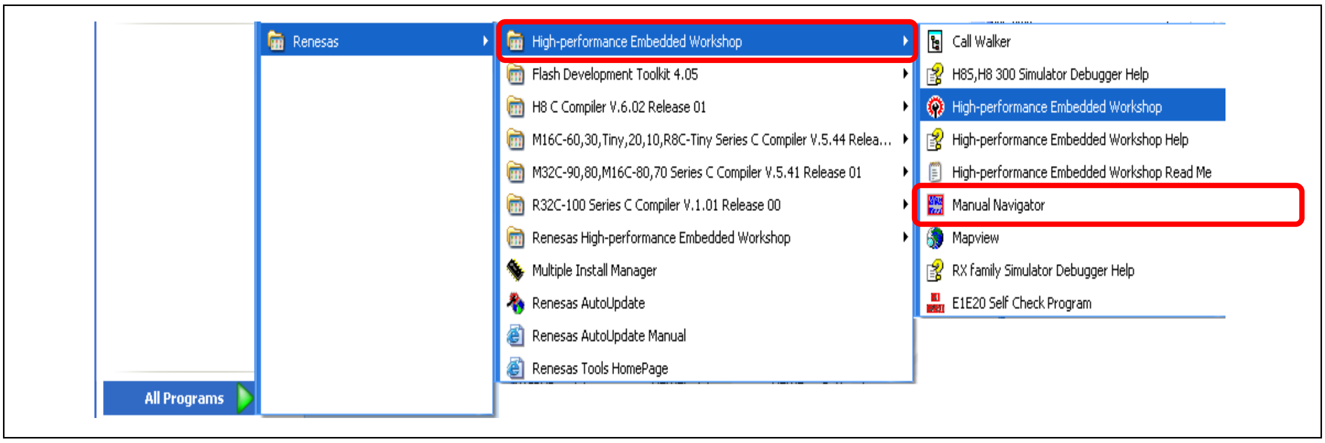

All Programs
Renesas
High-performance Embedded Workshop
Flash Development Toolkit 4.05
H8 C Compiler V.6.02 Release 01
M16C-60,30,Tiny,20,10,R8C-Tiny Series C Compiler V.5.44 Relea...
M32C-90,80,M16C-80,70 Series C Compiler V.5.41 Release 01
R32C-100 Series C Compiler V.1.01 Release 00
Renesas High-performance Embedded Workshop
Multiple Install Manager
Renesas AutoUpdate
Renesas AutoUpdate Manual
Renesas Tools HomePage
Call Walker
H8S,H8 300 Simulator Debugger Help
High-performance Embedded Workshop
High-performance Embedded Workshop Help
High-performance Embedded Workshop Read Me
Manual Navigator
Mapview
RX family Simulator Debugger Help
E1E20 Self Check Program

Figure 5.1 Invoking the Manual Navigator

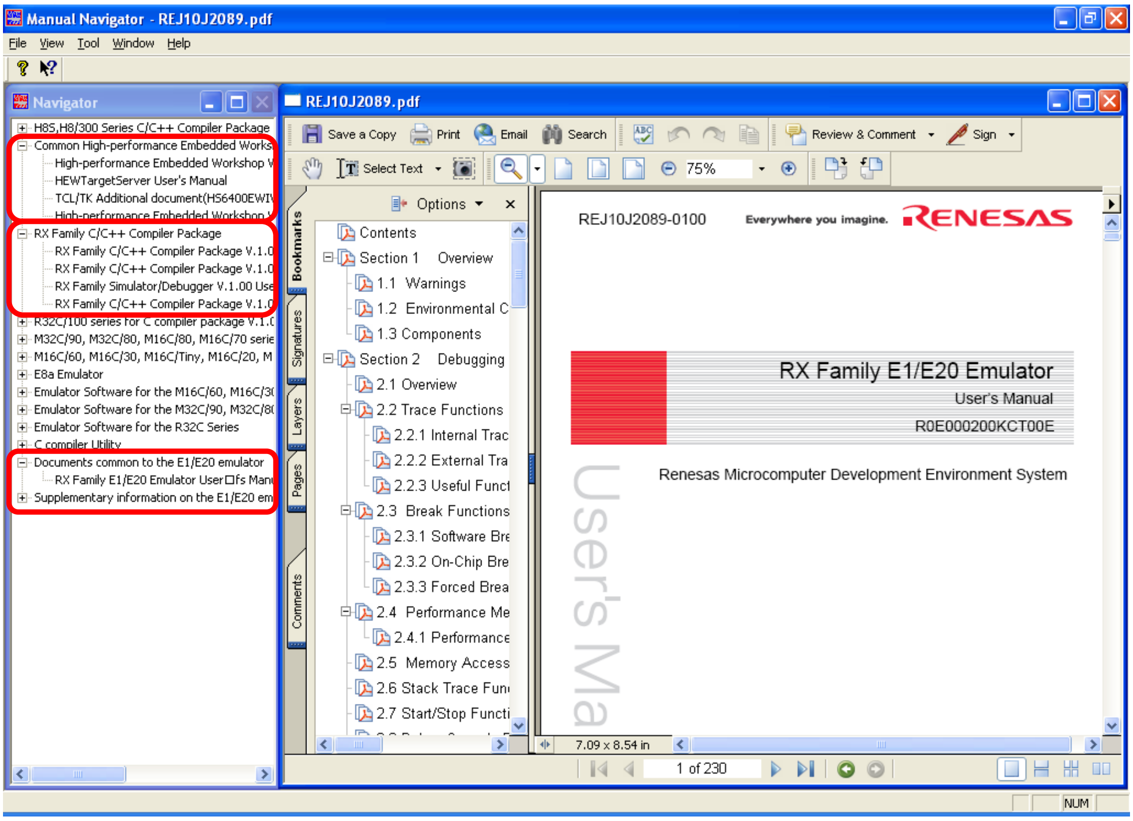

Manual Navigator - REJ10J2089.pdf
File View Tool Window Help
Navigator
H8S,H8/300 Series C/C++ Compiler Package
Common High-performance Embedded Works
High-performance Embedded Workshop V
HEWTargetServer User's Manual
TCL/TK Additional document(HS6400EWI
High-performance Embedded Workshop
RX Family C/C++ Compiler Package
RX Family C/C++ Compiler Package V.1.0
RX Family C/C++ Compiler Package V.1.0
RX Family Simulator/Debugger V.1.00 Use
RX Family C/C++ Compiler Package V.1.0
R32C/100 series for C compiler package V.1.0
M32C/90, M32C/80, M16C/80, M16C/70 serie
M16C/60, M16C/30, M16C/Tiny, M16C/20, M
E8a Emulator
Emulator Software for the M16C/60, M16C/30
Emulator Software for the M32C/90, M32C/80
Emulator Software for the R32C Series
C compiler Utility
Documents common to the E1/E20 emulator
RX Family E1/E20 Emulator User□fs Manu
Supplementary information on the E1/E20 em
REJ10J2089.pdf
Save a Copy
Print
Email
Search
Review & Comment
Sign
Select Text
75%
Options
Bookmarks
Signatures
Layers
Pages
Comments
Contents
Section 1 Overview
1.1 Warnings
1.2 Environmental C
1.3 Components
Section 2 Debugging
2.1 Overview
2.2 Trace Functions
2.2.1 Internal Trac
2.2.2 External Tra
2.2.3 Useful Funct
2.3 Break Functions
2.3.1 Software Bre
2.3.2 On-Chip Bre
2.3.3 Forced Brea
2.4 Performance Me
2.4.1 Performance
2.5 Memory Access
2.6 Stack Trace Fun
2.7 Start/Stop Functi
REJ10J2089-0100
Everywhere you imagine.
RENESAS
RX Family E1/E20 Emulator
User's Manual
R0E000200KCT00E
Renesas Microcomputer Development Environment System
User's Ma
7.09 x 8.54 in
1 of 230
NUM

Figure 5.2 Manual Navigator Screen

Also see the following pages for other useful information such as technical updates and restrictions.

http://www.renesas.com/e1	On-chip debugging emulator E1
http://www.renesas.com/hew	High-performance Embedded Workshop
http://www.renesas.com/rx_c	RX family C/C++ compiler package

5.2 Evaluation Version of the RX Family C/C++ Compiler Package

5.2.1 Conditions on Usage of the Evaluation-Version C Compiler

A free evaluation version of the C compiler is available so that you can evaluate the performance of the compiler before purchasing the official product. Note that the following conditions apply.

- After the evaluation-version software has been installed, there are no functional restrictions within 60 days from the day of the first building.
- From the 61st day, the maximum linkage size is limited to 128 Kbytes. The following message appears in the output window once 60 days have elapsed.

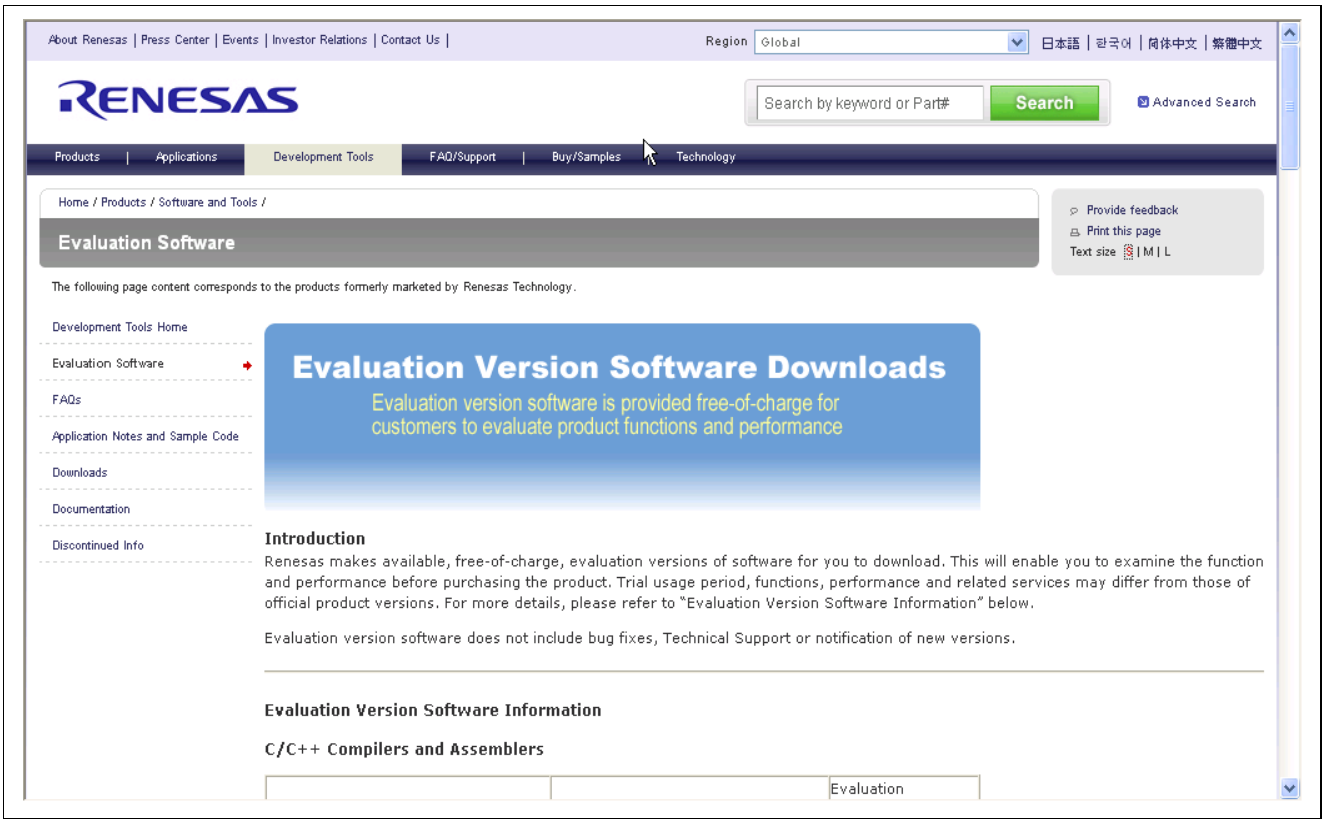
```
Software license problem: Duration of Trial License of UNKNOWN is exhausted.
Maximum link size limited to 128KB code+data.
```

5.2.2 Downloading the Evaluation-Version C Compiler

The evaluation-version C compiler can be downloaded from the following Web page.

http://www.renesas.com/tool_evaluation

To evaluate the compiler for RX MCUs, download the RX family C/C++ compiler package.


About Renesas | Press Center | Events | Investor Relations | Contact Us |
Region Global
日本語 | 한국어 | 简体中文 | 繁體中文
RENESAS
Search by keyword or Part#
Search
Advanced Search
Products | Applications | Development Tools | FAQ/Support | Buy/Samples | Technology
Home / Products / Software and Tools /
Provide feedback
Print this page
Text size S | M | L
Development Tools Home
Evaluation Software
FAQs
Application Notes and Sample Code
Downloads
Documentation
Discontinued Info
Evaluation

Evaluation Software

The following page content corresponds to the products formerly marketed by Renesas Technology.

Evaluation Version Software Downloads

Evaluation version software is provided free-of-charge for customers to evaluate product functions and performance

Introduction

Renesas makes available, free-of-charge, evaluation versions of software for you to download. This will enable you to examine the function and performance before purchasing the product. Trial usage period, functions, performance and related services may differ from those of official product versions. For more details, please refer to "Evaluation Version Software Information" below.

Evaluation version software does not include bug fixes, Technical Support or notification of new versions.

Evaluation Version Software Information

C/C++ Compilers and Assemblers

Figure 5.3 Downloading the Evaluation-Version Software

5.3 Information on the E20 Emulator

The E20 emulator has broader functionality than the E1.

The large-capacity trace memory in the E20 allows tracing of up to 2-M branches or cycles. The E20 also provides various powerful debugging functions including realtime RAM monitoring, which utilizes the acquired trace data to provide reference to the contents of memory in realtime.

Table 5.1 Differences between the E1 and E20

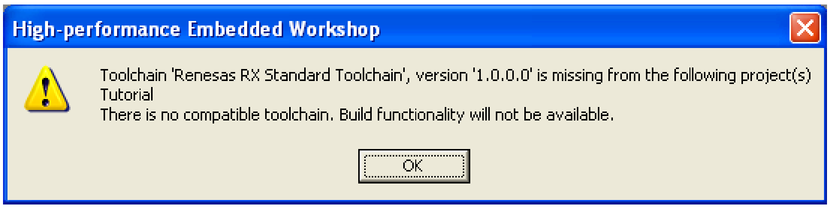
Emulator Functions		E1 (R0E000010KCE00)	E20 (R0E000200KCT00)
Events	Address execution	8 channels	
	Operand access	4 channels in total (on the CPU bus). 3 channels: Address/data masking is possible. 1 channel: Address or range/data masking is possible. The unit (1, 2, or 4 bytes) and type (R, W, R/W) of access are specifiable.	
	Event combination	OR, AND (cumulative), or sequential. An 8-bit pass counter is specifiable for a single event.	
Break	Software break	256 channels	
	On-chip break	Events are used.	
Trace	Capacity	256 branches or cycles	Minimum of 2-M branches or cycles
	Data type for tracing	Branch or operand access (on the CPU bus)	
	Data trace conditions	Events are used.	
	Conditions to start or stop tracing	Occurrence of a specified event or stopping the program	
	Display of trace data	Disassembly and C source modes are available. Backward stepping is also possible in C source mode.	
RAM monitoring	Auto-updating of the display in the [Memory] window	Possible (using a cycle-stealing transfer mode; execution stops for several cycles every time one word of data is read or written)	
	Realtime RAM monitoring	Not possible	Maximum of 4 Kbytes (still under development): Data and most recent access Access by the DMAC or DTC is not detectable.
Performance measurement		Two 32-bit counters or one 64-bit counter (to count the number of cycles that have elapsed, number of cycles of processing for interrupts or other exceptions, number of interrupts and other exceptions accepted, number of valid instructions issued, or number of event-condition matches)	

For details on the E20 emulator, refer to the RX Family E1/E20 Emulator User’s Manual.

6. Frequently Asked Questions

(1) Why does the warning message "Toolchain 'Renesas RX Standard Toolchain', version 'x.xx.x' is missing" shown in Figure 6.1 appear after I load a workspace?

This message indicates that a compatible compiler has not been installed or the compiler version that was used to create the workspace is not present on the host computer.

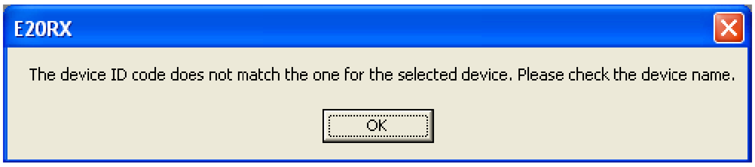


Toolchain 'Renesas RX Standard Toolchain', version '1.0.0.0' is missing from the following project(s)
Tutorial
There is no compatible toolchain. Build functionality will not be available.

Figure 6.1 Warning Message

(2) Why does the warning message "The device ID code does not match the one for the selected device name." shown in Figure 6.2 appear when I start up the debugger?

This message indicates that the device selected in the [Initial Settings] dialog box does not match the actual device on the user system connected to the E1 emulator. Select the correct device name in the [Initial Settings] dialog box.

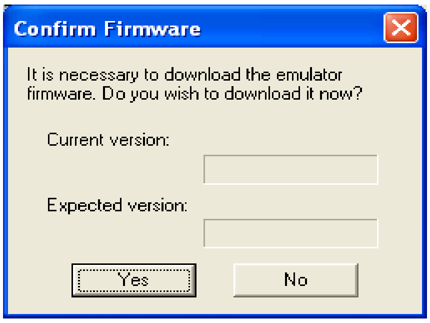


The device ID code does not match the one for the selected device. Please check the device name.

Figure 6.2 Warning Message

(3) Why does the [Confirm Firmware] dialog box shown in Figure 6.3 appear when I start up the debugger?

This message indicates that the firmware version corresponding to the emulator software and the firmware version in the E1 emulator are not the same. Click on the [Yes] button regardless of the version numbers shown in the dialog box.



It is necessary to download the emulator firmware. Do you wish to download it now?

Current version:

Expected version:

Figure 6.3 [Confirm Firmware] Dialog Box

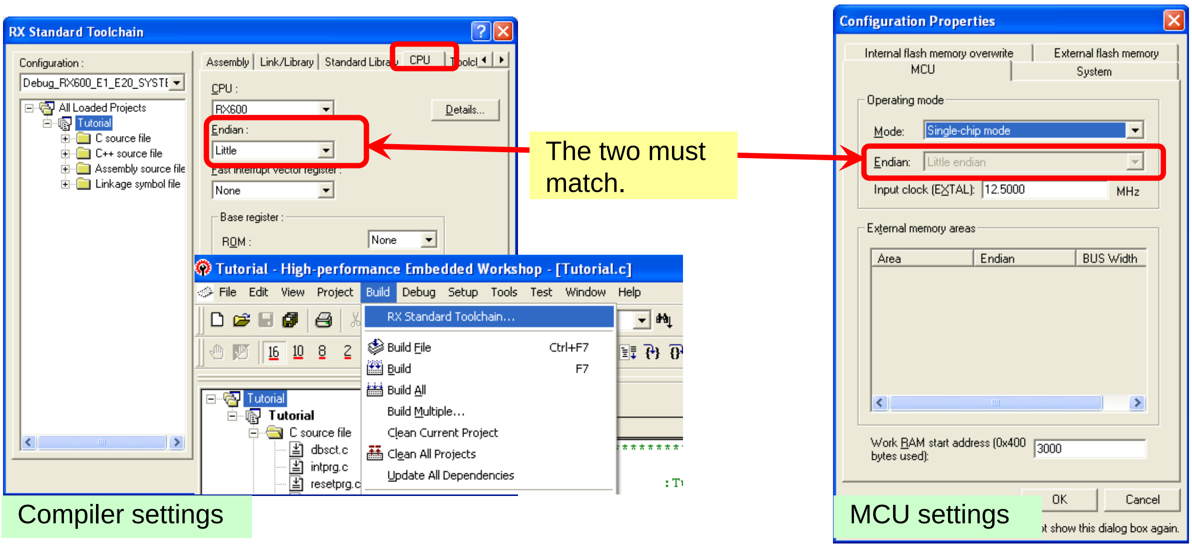
(4) What is the procedure when the downloaded program is not displayed correctly after I start up the debugger?

The endian is selectable for a RX600-family MCU. If the endian selected for the MCU does not match that selected when the program was created, the debugger cannot recognize the reset vector value of the downloaded program.

To check the endian selected when the program was created, select [Build]->[RX Standard Toolchain] in the High-performance Embedded Workshop to open the [RX Standard Toolchain] dialog box and see the [CPU] page (which shows compiler settings).

To check the endian selected for the MCU, see [Endian] on the [MCU] page of the [Configuration Properties] dialog box.

If the selections do not match, modify either of them.



The two must match.

Compiler settings

MCU settings

RX Standard Toolchain

Configuration : Debug_RX600_E1_E20_SYSTE

CPU : RX600

Endian : Little

Fast interrupt vector register : None

Base register : ROM : None

Configuration Properties

Operating mode

Mode: Single-chip mode

Endian: Little endian

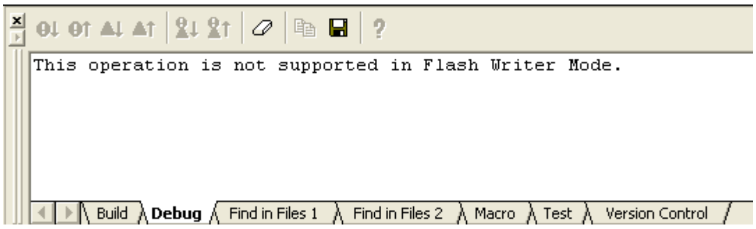
Input clock (EXTAL): 12.5000 MHz

Work RAM start address (0x400 bytes used): 3000

Figure 6.4 Checking That the Endians Match

(5) What is the procedure when I try to execute a debugging command but it is not accepted?

Check whether you have selected the flash programmer mode in the [Initial Settings] dialog box. If the message shown below is displayed in the output window, the emulator is operating in the flash programmer mode. To execute a debugging command, select the debugging mode instead.



```
This operation is not supported in Flash Writer Mode.
```

Figure 6.5 Display in the Output Window

(6) Since I installed the E1/E20 emulator software V.1.00 Release 00 on a host computer running Windows® XP, no other emulators are connectable to the host computer. How do I connect another emulator?

Follow the procedure below to check the operating state of USB driver.

Connect the emulator to the host computer and supply power to the emulator.

Start up the Windows' Device Manager to see the list of devices.

Open the properties for the Renesas' USB driver and click on the [General] tab.

If the "The drivers for this device need to be reinstalled." message is displayed in the [Device status] section, take the following steps to reinstall the USB driver.

1 Right-click on [Renesas E-Series USB Driver] under the [Renesas Emulator] category in the Device Manager and select [Delete] from the menu.
2 Turn off power to the emulator.*
3 Turn on power to the emulator.*
4 [Found New Hardware Wizard] automatically starts. Select [Install the software automatically (Recommended)] and click on the [Next] button.
5 [Found New Hardware Wizard] shows the "Please select the best match for your hardware from the list below." message.
6 When two or more inf files are displayed, select one and click on the [Next] button.
7 Check that [Found New Hardware Wizard] shows the "The wizard has finished installing the software for: Renesas E-Series USB Driver" message. Click on the [Finish] button.

Note: Turning power to the emulator on and off
Most on-chip emulators can only be supplied with power while they are connected to the host computer via a USB interface cable. To turn power to such emulators on or off, connect or disconnect the USB interface cable.

This note applies to the emulators given below. That is, the only exception is the E8a.

— E10A-USB
— E1
— E20
— E8
— E7
— E10T-USB
— E100
— E30A
— E200F

(7) Other Questions
The most common questions and answers about Renesas products, including the E1, are given as "FAQs" on the Renesas Web site. Enter the E1 emulator page and click on "FAQs".

Homepage and Contact Information

Renesas Electronics’ Homepage
http://www.renesas.com/

Contact
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 07, 2010	—	First edition issued

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.
2. 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 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. 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; anti-crime 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 majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

SALES OFFICES **Renesas Electronics Corporation** http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.0