

CubeSuite+ V1.01.00

Integrated Development Environment

User's Manual: RX Debug

Target Device

RX Family

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (<http://www.renesas.com>).

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.

How to Use This Manual

This manual describes the role of the CubeSuite+ integrated development environment for developing applications and systems for RX family, and provides an outline of its features.

CubeSuite+ is an integrated development environment (IDE) for RX family, integrating the necessary tools for the development phase of software (e.g. design, implementation, and debugging) into a single platform.

By providing an integrated environment, it is possible to perform all development using just this product, without the need to use many different tools separately.

Readers	This manual is intended for users who wish to understand the functions of the CubeSuite+ and design software and hardware application systems.
Purpose	This manual is intended to give users an understanding of the functions of the CubeSuite+ to use for reference in developing the hardware or software of systems using these devices.
Organization	This manual can be broadly divided into the following units.

CHAPTER 1 GENERAL
CHAPTER 2 FUNCTIONS
APPENDIX A WINDOW REFERENCE
APPENDIX B INPUT/OUTPUT FUNCTIONS
APPENDIX C INDEX

How to Read This Manual	It is assumed that the readers of this manual have general knowledge of electricity, logic circuits, and microcontrollers.
--------------------------------	--

Conventions	Data significance: Higher digits on the left and lower digits on the right
	Active low representation: \overline{XXX} (overscore over pin or signal name)
	Note: Footnote for item marked with Note in the text
	Caution: Information requiring particular attention
	Remark: Supplementary information
	Numeric representation: Decimal ... XXXX
	Hexadecimal ... 0xFFFF

Related Documents

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Document Name		Document No.
CubeSuite+ Integrated Development Environment User's Manual	Start	R20UT0727E
	V850 Design	R20UT0549E
	R8C Design	R20UT0550E
	RL78 Design	R20UT0728E
	78K0R Design	R20UT0547E
	78K0 Design	R20UT0546E
	RX Coding	R20UT0767E
	V850 Coding	R20UT0553E
	Coding for CX Compiler	R20UT0825E
	R8C Coding	R20UT0576E
	RL78, 78K0R Coding	R20UT0729E
	78K0 Coding	R20UT0782E
	RX Build	R20UT0768E
	V850 Build	R20UT0557E
	Build for CX Compiler	R20UT0826E
	R8C Build	R20UT0575E
	RL78, 78K0R Build	R20UT0730E
	78K0 Build	R20UT0783E
	RX Debug	This manual
	V850 Debug	R20UT0734E
	R8C Debug	R20UT0770E
	RL78 Debug	R20UT0733E
	78K0R Debug	R20UT0732E
	78K0 Debug	R20UT0731E
	Analysis	R20UT0735E
	Message	R20UT0736E

Caution The related documents listed above are subject to change without notice. Be sure to use the latest edition of each document when designing.

All trademarks or registered trademarks in this document are the property of their respective owners.

TABLE OF CONTENTS

CHAPTER 1 GENERAL ... 8

- 1.1 Summary ... 8
- 1.2 Features ... 8

CHAPTER 2 FUNCTIONS ... 9

- 2.1 Overview ... 9
- 2.2 Preparation before Debugging ... 12
 - 2.2.1 Confirm the connection to a host machine ... 12
- 2.3 Configuration of Operating Environment of the Debug Tool ... 14
 - 2.3.1 Select the debug tool to use ... 14
 - 2.3.2 [E1] ... 15
 - 2.3.3 [E20] ... 26
 - 2.3.4 [Simulator] ... 38
- 2.4 Connect to/Disconnect from the Debug Tool ... 44
 - 2.4.1 Connect the debug tool to CubeSuite+ ... 44
 - 2.4.2 Disconnect the debug tool from CubeSuite+ ... 44
 - 2.4.3 Connect the debug tool to CubeSuite+ using hot plug-in [E1(JTAG)] [E20(JTAG)] ... 45
- 2.5 Download and Upload ... 47
 - 2.5.1 Execute downloading ... 47
 - 2.5.2 An applied method of download ... 49
 - 2.5.3 Executing an upload ... 58
- 2.6 Displaying and Changing Programs ... 60
 - 2.6.1 Displaying source files ... 60
 - 2.6.2 Displaying the disassembled result ... 66
 - 2.6.3 Executing a build in parallel with other processes ... 69
 - 2.6.4 Performing line assembly ... 70
- 2.7 Usage of PIC/PID Function ... 72
 - 2.7.1 Changing the allocation of a load module using the PIC/PID function ... 73
- 2.8 Setting Overlay Sections ... 75
 - 2.8.1 Selecting the priority section ... 76
- 2.9 Execute Programs ... 78
 - 2.9.1 Reset microcontroller (CPU) ... 78
 - 2.9.2 Execute programs ... 78
 - 2.9.3 Execute programs in steps ... 80
 - 2.9.4 Execute a specified routine [E1] [E20] ... 81
- 2.10 Stop Programs (Break) ... 83
 - 2.10.1 Stop the program manually ... 83
 - 2.10.2 Stop the program at the arbitrary position (breakpoint) ... 83
 - 2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20] ... 85
 - 2.10.4 Stop the program with the access to variables/I/O registers ... 87
 - 2.10.5 Set multiple break events in combination (Combination break) [E1] [E20] ... 92

2.10.6	Other break factors ...	93
2.11	Displaying and Changing Memory, Registers, and Variables ...	95
2.11.1	Displaying and changing memory contents ...	95
2.11.2	Displaying and changing the CPU registers ...	106
2.11.3	Displaying and changing the I/O registers ...	108
2.11.4	Displaying and changing global and static variables ...	111
2.11.5	Displaying and changing local variables ...	111
2.11.6	Displaying and changing watch expressions ...	113
2.12	Display Function Call Information from the Stack ...	119
2.12.1	Display call stack information ...	119
2.13	Collecting an Execution History ...	121
2.13.1	Setting up a trace operation ...	121
2.13.2	Collecting an execution history up to a halt ...	125
2.13.3	Collecting an execution history in a section ...	125
2.13.4	Collecting an execution history only when conditions are met ...	130
2.13.5	Displaying an execution history ...	131
2.13.6	Clearing the trace memory ...	133
2.13.7	Searching for trace data ...	134
2.13.8	Saving the displayed content of an execution history ...	139
2.14	Measuring the Execution Time ...	141
2.14.1	Setting the timer measurement operation [E1] [E20] ...	141
2.14.2	Measuring execution time from start to stop ...	141
2.14.3	Measuring execution time in a section [E1] [E20] ...	143
2.14.4	Range of measurable time ...	147
2.15	Measure Coverage [Simulator] ...	148
2.15.1	Configure the coverage measurement ...	148
2.15.2	Display coverage measurement results ...	148
2.16	Set an Action into Programs ...	151
2.16.1	Insert printf ...	151
2.16.2	Insert an interrupt event [Simulator] ...	153
2.17	Event Management ...	156
2.17.1	Changing states of setting (Enabled/Disabled) ...	157
2.17.2	Displaying only a specific type of event ...	158
2.17.3	Jump to an event address ...	158
2.17.4	Editing detailed settings of events ...	158
2.17.5	Deleting events ...	165
2.17.6	Entering comments for events ...	166
2.17.7	Points to note regarding event setting ...	166
2.18	Setting Up the Hook Process ...	169
2.19	Using the Debug Console ...	172
2.20	About Input Value ...	178
2.20.1	Input rule ...	178
2.20.2	Symbol name completion function ...	181
2.20.3	Icons for invalid input ...	182

APPENDIX A WINDOW REFERENCE ... 183

A.1 Description ... 183

APPENDIX B INPUT/OUTPUT FUNCTIONS ... 406

B.1 Standard Input/Output and File Input/Output ... 406

APPENDIX C INDEX ... 421

CHAPTER 1 GENERAL

CubeSuite+ is an integrated developing environment platform for the RX family, V850 microcontroller, RL78 family, 78K0R microcontroller and 78K0 microcontroller.

CubeSuite+ comprehensively supports the operations required for program development such as designing, coding, building, debugging, and flash programming.

In this manual, the debugging is explained out of those operations needed for the program development.

Among them, this manual focuses on the description of debugging the outline of which is provided in this chapter.

1.1 Summary

You can efficiently debug the program developed for the RX family, using the debugger provided by CubeSuite+.

1.2 Features

The following are the features of the debugger provided by CubeSuite+.

- Connectable to various debugging tools

Usable in combination with the on-chip debugging emulator (E1/E20) and the simulator, it provides improved efficiency in program development.

- Mixed display of C/C++ source text and disassembled text

The C/C++ source text and the disassembled text can be displayed together on the same panel.

- Source level debugging and instruction level debugging

C/C++ source program can be debugged either at the source or the instruction level.

- Data flash memory programming

When the selected microcontroller is provided with a data flash memory, you can display or edit the contents of the data flash memory using the same access method as in other normal memory operations (not including the simulator).

- Real-time display updating function

The values of memory, registers and variables are automatically updated not only when the program is stopped, but also in execution.

- Saving/restoring the debugging environment

The debugging environment such as breakpoints, event configuration information, file download information, display condition/position of the panel, etc. can be saved.

CHAPTER 2 FUNCTIONS

This chapter describes the debugging process using CubeSuite+ as well as main debugging functions.

2.1 Overview

Following shows the basic sequence of program debugging using CubeSuite+.

(1) Start CubeSuite+

Launch CubeSuite+ from the [Start] menu of Windows®.

Remark For details on "Start CubeSuite+", see "CubeSuite+ Start".

(2) Set a project

Create a new project, or load the existing one.

Remark For details on "Set a project", see "CubeSuite+ Start".

(3) Create a load module

Once you are finished with the setting of the active project and the build, execute the build to create a load module.

Remark For details on "Create a load module" with CC-RX, see "CubeSuite+ Build".

(4) Confirm the connection to a host machine

Connect the debug tool (E1, E20 or Simulator) to be used to a host machine.

(5) Select the debug tool to use

Select the debug tool to be used in a project.

Remark The selectable debug tool differs depending on the microcontroller type to be used in a project.

(6) Configure the operating environment for the debug tool

Configure the operating environment for the debug tool selected in step (5).

- [E1]
- [E20]
- [Simulator]

(7) Connect the debug tool to CubeSuite+

Connect the debug tool to CubeSuite+ to start communication.

(8) Execute downloading

Download the load module created in step (3) to the debug tool.

(9) Displaying source files

Display the contents of the downloaded load module (source files) on the [Editor panel](#) or [Disassemble panel](#).

(10) Execute programs

Execute the program using the operation method that matches your purpose.

If you wish to stop the program at the arbitrary position, set a breakpoint/break event^{Note} before executing the program (see "2.10.2 Stop the program at the arbitrary position (breakpoint)"/"2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20]").

Note These functions are implemented by setting events to the debug tool used.

See "2.17.7 Points to note regarding event setting", when you use events.

(11) Stop the program manually

Stop the program currently being executed.

Note that if a breakpoint/break event has been set in steps (10), the program execution will be stopped automatically when the condition of the breakpoint/break event is met.

(12) Check the result of the program execution

Check the following information that the debug tool acquired by the program execution.

- [Displaying and Changing Memory, Registers, and Variables](#)
- [Display Function Call Information from the Stack](#)
- [Collecting an Execution History](#)^{Note}
- [Measuring the Execution Time](#)^{Note}
- [Measure Coverage \[Simulator\]](#)

Note These functions are implemented by setting events to the debug tool used.

See "2.17.7 Points to note regarding event setting", when you use events.

Debug the program, repeating steps (9) to (12) as required.

Note that if the program is modified during debugging, steps (3) and (8) also should be repeated.

Remarks 1. Other than the above, you can also check the result of the program execution by using the following functions.

- [Set an Action into Programs](#)
 - [Setting Up the Hook Process](#)
 - [Using the Debug Console](#)
2. The acquired information can be saved to a file.
- [Saving the displayed contents of disassembled results](#)
 - [Saving displayed memory contents](#)
 - [Saving the displayed CPU register contents](#)
 - [Saving the displayed I/O register contents](#)
 - [Saving the displayed contents of local variables](#)
 - [Saving the displayed contents of watch expressions](#)
 - [Saving the displayed contents of call-stack information](#)
 - [Saving the displayed content of an execution history](#)

(13) Executing an upload

Save the program (the memory contents) to a file in the arbitrary format (e.g. Intel hex format, binary data format, etc.), as required.

(14) Disconnect the debug tool from CubeSuite+

Disconnect the debug tool from CubeSuite+ to terminate communication.

(15) Save the project file

Save the setting information of the project to the project file.

Remark For details on "Save the project file", see "CubeSuite+ Start".

2.2 Preparation before Debugging

This section describes the preparation to start debugging the created program.

2.2.1 Confirm the connection to a host machine

Connection examples for each debug tool are shown.

Note that the relationship between the microcontroller selected in the project and the connectable debug tools is as the following table.

Remark "(Serial)"/"(JTAG)" in the table below means that E1/E20 is used with FINE communications or JTAG communications.

Table 2-1. Relationship between Types of Microcontroller and Connectable Debug Tools

Microcontroller	Debug Tool				
	E1(Seria)	E1(JTAG)	E20(Serial)	E20(JTAG)	Simulator
RX610, RX621, RX62N, and RX62T Groups	-	✓	-	✓	✓
RX630, RX631, and RX63N Groups	✓	✓	✓	✓	✓
RX210 Group	✓	-	✓	-	✓

- (1) [E1]
- (2) [E20]
- (3) [Simulator]

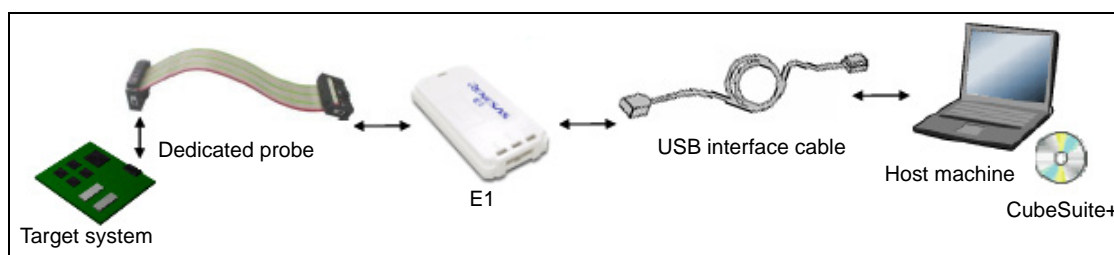
(1) [E1]

Connect a host machine and E1. If required, connect a target board, too.

You can make a choice between FINE communications [E1(Serial)] and JTAG communications [E1(JTAG)] as the communication method to the target system (see "2.3.1 Select the debug tool to use").

See E1 User's Manual for details on the connection method.

Figure 2-1. Connection Example [E1]



(2) [E20]

Connect a host machine and E20. If required, connect a target board, too.

You can make a choice between FINE communications [E20(Serial)] and JTAG communications [E20(JTAG)] as the communication method to the target system (see "2.3.1 Select the debug tool to use").

See E20 User's Manual for details on the connection method.

Figure 2-2. Connection Example [E20(JTAG)]

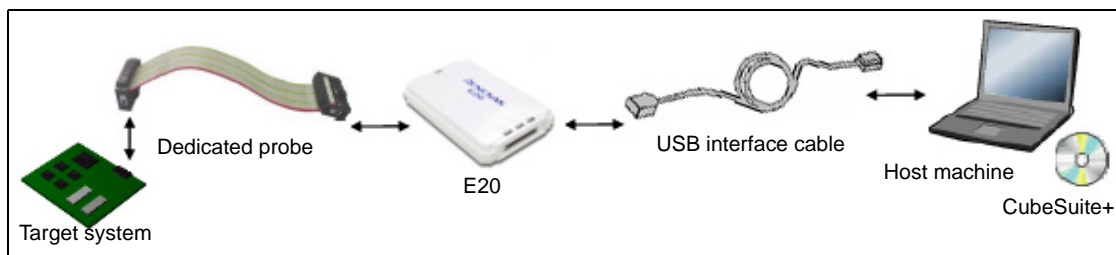
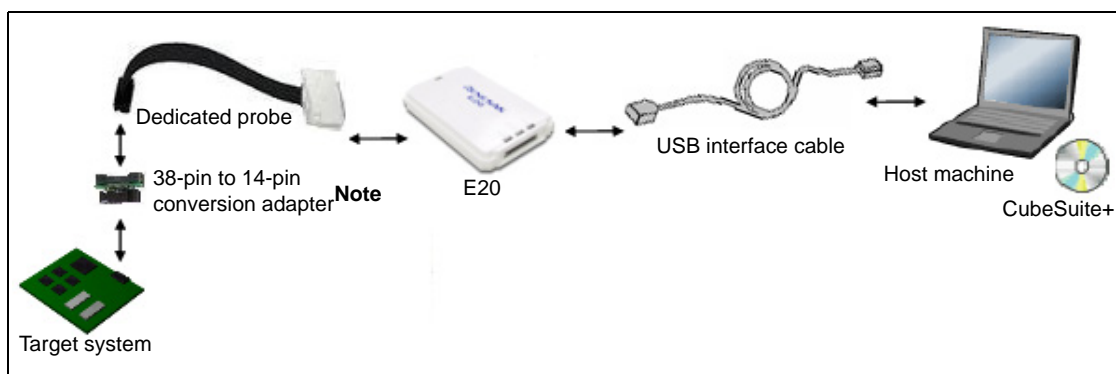


Figure 2-3. Connection Example [E20(Serial)]



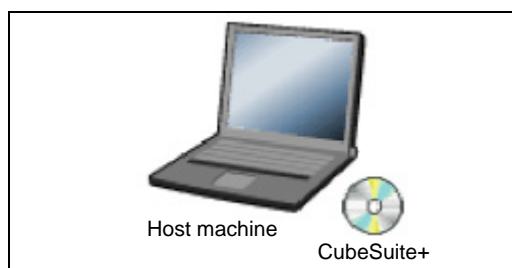
Note Use a 38-pin to 14-pin conversion adapter attached to the E20 emulator if you have selected FINE [E20 (Serial)] as the method of communication with the target system.

For more details on the connection using a 38-pin to 14-pin conversion adapter, see E20 User's Manual

(3) [Simulator]

A host machine is only needed for debugging (emulators are not needed).

Figure 2-4. Connection Example [Simulator]



2.3 Configuration of Operating Environment of the Debug Tool

This section describes the configuration of the operating environment for each debug tool.

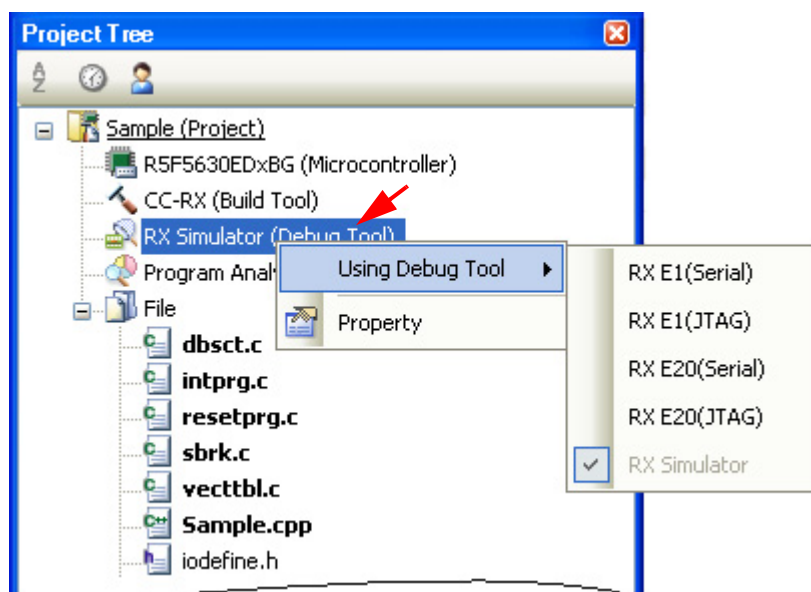
2.3.1 Select the debug tool to use

You can configure the operating environment in the [Property panel](#) corresponding to the debug tool to use.

Therefore, first, select the debug tool to be used in a project (the debug tool to be used can be specified in the individual projects).

To select or switch the debug tool, use the context menu shown by right clicking on the [*Microcontroller type Debug tool name (Debug Tool)*] node on the [Project Tree panel](#).

Figure 2-5. Select/Switch Debug Tool to Use



Caution The context menu items displayed differ depending on the microcontroller selected in the project (see "[Table 2-1. Relationship between Types of Microcontroller and Connectable Debug Tools](#)").

Remark [E1] [E20]

Select [*Microcontroller type E1(Serial)*]/[*Microcontroller type E20(Serial)*] to perform FINE communications between E1/E20 and the target system.

Select [*Microcontroller type E1(JTAG)*]/[*Microcontroller type E20(JTAG)*] to perform JTAG communications between E1/E20 and the target system.

If the [Property panel](#) is already open, click the [*Microcontroller type Debug tool name (Debug Tool)*] node again. The view switches to the Property panel of the selected debug tool.

If the Property panel is not open, double-click the above mentioned node to open the corresponding Property panel.

2.3.2 [E1]

Configure the operating environment on the [Property panel](#) below when using E1.

Note that the contents of the [Property panel](#) differ depending on the communication method (FINE communications [E1(Serial)] or JTAG communications [E1(JTAG)]) between E1 and the target system.

Figure 2-6. Property Panel [E1(Serial)]

The screenshot shows the 'Property' window for 'RX E1(Serial)'. It contains several expandable sections with various configuration options.

Section	Property	Value
Internal ROM/RAM	Size of internal ROM[KBytes]	2048
	Size of internal RAM[KBytes]	128
	Size of DataFlash memory[KBytes]	32
Clock	Main clock source	EXTAL
	Main clock frequency[MHz]	
	Allow changing of the clock source on writing internal flash memory	No
Connection with Emulator	Emulator serial No.	E1: XXXXXXXXXX
Connection with Target Board	Power target from the emulator.(MAX 200mA)	No
	Communications method	FINE
	FINE baud rate[bps]	2000000
Flash	Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
	ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
	Work RAM start address	HEX 1000
	Work RAM size[bytes]	1024
Operating Modes of CPU	Mode pins setting	Single-chip mode
	Register setting	Single-chip mode
	Endian	Little-endian data
External Flash	External flash definition file	[4]
Size of internal ROM[KBytes] Size of the internal ROM		

At the bottom, there are tabs for 'Connect Settings', 'Debug Tool Settings', 'Download File Settings', and 'Hook Transaction Settings'. 'Connect Settings' is currently selected.

Figure 2-7. Property Panel [E1(JTAG)]

Internal ROM/RAM	
Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No

Connection with Emulator	
Emulator serial No.	E1: XXXXXXXXX

Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	JTAG
JTAG clock[MHz]	16.5

Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024

Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data

External Flash	
External flash definition file	[4]

Internal ROM/RAM

Connect Settings | Debug Tool Settings | Download File Settings | Hook Transaction Settings

Follow the steps below by selecting the corresponding tab on the [Property panel](#).

- (1) [\[Connect Settings\] tab](#)
- (2) [\[Debug Tool Settings\] tab](#)
- (3) [\[Download File Settings\] tab](#)
- (4) [\[Hook Transaction Settings\] tab](#)

(1) [\[Connect Settings\] tab](#)

In the [\[Connect Settings\] tab](#), you configure the connection with the debug tool for each one of the following categories.

- (a) [\[Internal ROM/RAM\]](#)
- (b) [\[Clock\]](#)
- (c) [\[Connection with Emulator\]](#)
- (d) [\[Connection with Target Board\]](#)
- (e) [\[Flash\]](#)
- (f) [\[Operating Modes of CPU\]](#)
- (g) [\[External Flash\]](#)

(a) [\[Internal ROM/RAM\]](#)

The configuration of internal ROM/RAM is displayed in this category.

Figure 2-8. [Internal ROM/RAM] Category

Internal ROM/RAM	
Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32

<1> [Size of internal ROM[KBytes]]

The internal ROM size to emulate is displayed (unit: Kbytes).

You cannot change the value of this property.

<2> [Size of internal RAM[Bytes]]

The internal RAM size to emulate is displayed (unit: bytes).

You cannot change the value of this property.

<3> [Size of DataFlash memory[KBytes]]

The data flash memory size is displayed (unit: Kbytes).

If the currently selected microcontroller does not incorporate the data flash, [0] is displayed.

You cannot change the value of this property.

(b) [Clock]

You can configure the clock in this category.

Caution You cannot change the property in this category while connected to E1.

Figure 2-9. [Clock] Category [HOCO]

Clock	
Main clock source	HOCO
Allow changing of the clock source on writing internal flash memory	No

Figure 2-10. [Clock] Category [EXTAL]

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Allow changing of the clock source on writing internal flash memory	No

<1> [Main clock source]

Select EXTAL frequency or internal HOCO as the main clock source. [EXTAL] will be displayed for microcontrollers with no internal HOCO.

<2> [Main clock frequency [MHz]]

Specify the main clock frequency (before multiplier).

Specify EXTAL frequency by directly entering a number between 0.0001 and 99.9999 (MHz). The entered value will be truncated to 5 decimal places. If the value is out of the specifiable range, it will be rounded to 0.0001 (when 0 or below) or to 99.9999 (when 100 or above).

This property is displayed only when you have selected [EXTAL] in the [Main clock source] property.

<3> [Allow changing of the clock source on writing internal flash memory]

Specify whether to allow a debugger to operate the clock while the internal flash memory is being rewritten. When [Yes] is selected, any clock setting outside the guaranteed range of microcontroller operation will be modified by E1 to be in accord with the specified limits for reprogramming internal flash memory.

(c) [Connection with Emulator]

You can configure the connection between E1 and the host machine in this category.

Caution You cannot change the property in this category while connected to E1.

Figure 2-11. [Connection with Emulator] Category

☐ Connection with Emulator	
Emulator serial No.	E1: XXXXXXXXXX

<1> [Emulator serial No.]

Serial numbers of all connected E1 emulators are displayed in the drop-down list.
Select the one to be connected to the target system.
The drop-down list is updated every time it is used.

(d) [Connection with Target Board]

You can configure the connection between E1 and the target board in this category.

Figure 2-12. [Connection with Target Board] Category [E1(Serial)]

☐ Connection with Target Board	
Power target from the emulator.(MAX 200mA)	Yes
Supply voltage	3.3V
Communications method	FINE
FINE baud rate[bps]	2000000

Figure 2-13. [Connection with Target Board] Category [E1(JTAG)]

☐ Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	JTAG
JTAG clock[MHz]	16.5

<1> [Power target from the emulator. (MAX 200mA)]

Specify whether to supply power to the target system from E1.
Select [Yes] to supply power to the target system ([No] is selected by default).

Caution This property cannot be changed while connected to E1.

<2> [Supply voltage]

Specify the power voltage supplied to the target system from the following drop-down list.
This property appears only when the [Power target from the emulator. (MAX 200mA)] property is set to [Yes].
The voltage value that can be specified differs depending on the type of the microcontroller.

Caution This property cannot be changed while connected to E1.

<3> [Communications method]

Displays the method of communication used by the E1 emulator for communicating with the microcontroller on the target system. Specifying [RX E1(Serial)] for a debug tool in the [Project Tree panel](#) will display [FINE] in this property, and specifying [RX E1(JTAG)] will display [JTAG].

You cannot change the value of this property.

For the details of debug tool selection, see "[2.3.1 Select the debug tool to use](#)".

<4> [JTAG clock[MHz]]

From the drop-down list, select the baud rate (JTAG clock) to be used by the E1 emulator for communicating with the microcontroller on the target system.

This property is displayed only when [JTAG] is selected in the [\[Communications method\]](#) property.

The following baud rate is displayed in the drop-down list.

- 16.5 (default), 12.38, 6.188, 3.094, 1.547

Cautions 1. This property cannot be changed while connected to E1.

2. Depending on the length or the method of JTAG signal wiring on the target system, it may not be possible to communicate using the selected JTAG clock. In such a case, reducing the JTAG clock may achieve successful communication.

<5> [FINE baud rate[bps]]

From the drop-down list, select the baud rate (FINE baud rate) to be used by the E1 emulator for communicating with the microcontroller on the target system.

This property is displayed only when [FINE] is selected in the [\[Communications method\]](#) property.

The following baud rate is displayed in the drop-down list.

- 2000000 (default), 750000, 500000, 250000

Cautions 1. This property cannot be changed while connected to E1.

2. Depending on the length or the method of FINE signal wiring on the target system, it may not be possible to communicate using the selected FINE baud rate. In such a case, reducing the JTAG clock may achieve successful communication.

(e) [Flash]

You can configure the flash memory rewriting in this category.

Caution This property cannot be changed while connected to E1.

Figure 2-14. [Flash] Category

Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024

<1> [Input mode of ID code]

Specify a mode for the input of ID code.

Caution This property cannot be changed while connected to E1.

<2> [ID code]

Specify the ID code to use when reading the code from the memory.

If you have selected [Specify the ID code as a 32-digit hexadecimal] in the [\[Input mode of ID code\]](#) property, enter the ID code in a 32-digit hexadecimal number. If you have selected [Specify the ID code as an ASCII code within 16 characters], enter the ID code using maximum 16 ASCII characters.

- Cautions**
1. This property cannot be changed while connected to E1.
 2. If the ID code entered in ASCII characters is shorter than 16 characters, the unused space will be padded with 0.

<3> [Work RAM start address]

Specify the location address of the work RAM to be used by the debugger.

The size of work RAM to be used by the debugger firmware at a designated RAM location address is indicated in [\[Work RAM size\[bytes\]\]](#) property.

- Cautions**
1. This property cannot be changed while connected to E1.
 2. This area can also be used by the user program since the memory contents are swapped and restored. However, you cannot designate this area as the source or destination of DMA or DTC function transfer.

<4> [Work RAM size[bytes]]

Displays the size of the work RAM to be used by the debugger.

(f) [Operating Modes of CPU]

In this category, you configure the operating mode of the microcontroller to be emulated.

Caution This property cannot be changed while connected to E1.

Figure 2-15. [Operating Modes of CPU] Category

Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data

<1> [Mode pins setting]

Specify the operating mode set by the mode pin of the microcontroller.

<2> [Register setting]

Specify the operating mode to be set by the register.

The operating mode that can be specified depends on the type of the microcontroller.

<3> [Endian]

Display the project endian. Acquires endian information from the project and displays its value. Can be selected only when the debug tool is disconnected.

(g) [External Flash]

In this category, you can configure external flash.

The settings in this category are required when downloading to an external flash memory. For more details regarding property setting, see "(5) [Downloading files to external flash memory \[E1\] \[E20\]](#)".

Caution You cannot change the property in this category while connected to E1.

(2) [Debug Tool Settings] tab

In the [Debug Tool Settings] tab, you configure the basic settings of the debug tool for each one of the following categories.

- (a) [Memory]
- (b) [Access Memory While Running]
- (c) [Break]
- (d) [System]
- (e) [Trace]
- (f) [Timer]

(a) [Memory]

You can configure the memory in this category.

Figure 2-16. [Memory] Category

Memory	
Memory mappings	[28]
[0]	On-chip RAM area
[1]	Reserved area
[2]	I/O registers area
[3]	Reserved area
[4]	I/O registers area
[5]	I/O registers area
[6]	Reserved area
[7]	Data flash area
[8]	Reserved area
[9]	Other memory area
[10]	Reserved area
[11]	I/O registers area
[12]	Reserved area
[13]	I/O registers area
[14]	Reserved area
[15]	External area (CS7)
[16]	External area (CS6)
[17]	External area (CS5)
[18]	External area (CS4)
[19]	External area (CS3)
[20]	External area (CS2)
[21]	External area (CS1)
[22]	Reserved area
[23]	Other memory area
[24]	Reserved area
[25]	Other memory area
[26]	Reserved area
[27]	On-chip ROM area
Verify on writing to memory	Yes

<1> [Memory mappings]

Current memory mapping status is displayed for each type of memory area.

This property displays only the number of memory areas.

Expanding [Memory mapping] property will display the following sub-items.

- [Memory type]

Indicates the memory type of the corresponding area.

Each memory type corresponds to the following areas.

On-chip ROM area	Program ROM and data flash
On-chip RAM area	Internal RAM
I/O registers area	Peripheral I/O register Divided into areas with different endians for display
External area(CS7/CS6/.../CS0)	External address space CS0 to CS7 are displayed separately
Other memory area	FCU-RAM, FCU firm, user boot
Reserved area	Areas other than those listed above

- [Start address]

Displays the starting address of the corresponding area.

- [End address]

Displays the ending address of the corresponding area.

- [Access width[bits]]

Displays the access width of the corresponding area.

When [Memory type] is an external area, the access width can only be changed when the debug tool is disconnected.

- [Endian]

Displays the endians of the external area and the I/O register area.

When [Memory type] is an external area, the endian can only be changed when the debug tool is disconnected.

Caution Connecting to a debug tool (see "2.4.1 Connect the debug tool to CubeSuite+") will display details for each memory type.

<2> [Verify on writing to memory]

Specify whether to perform a verify check when the memory value is initialized from the drop-down list.

Select [Yes] to perform verification after download or when values are changed in the [Watch panel](#)/
[Memory panel](#).

(b) [Access Memory While Running]

You can configure the memory access while executing a program in this category.

The settings of this category are required when using the real-time display update function. See "(4) [Displaying and changing memory contents during program execution](#)" for details on the real-time display update function.

Figure 2-17. [Access Memory While Running] Category

<input checked="" type="checkbox"/> Access Memory While Running	
Access by stopping execution	No
Update display during the execution	Yes
Display update interval[ms]	500

<1> [Access by stopping execution]

Specify from the drop-down list whether to allow access to the memory area while executing a program.
Select [Yes] to allow access ([No] is selected by default).

<2> [Update the display during execution]

Specify whether to update the display in the [Watch panel/Memory panel](#) while executing a program.
Select [Yes] to update the display (default).

<3> [Update interval[ms]]

This property is valid only when the [\[Update the display during execution\]](#) property is set to [Yes].
Specify the interval in 100ms unit to update the contents in the [Watch panel/Memory panel](#) display while executing a program.
Directly enter the Integer number between 100 and 65500 (rounding up the fractions less than 100ms) ([500] is specified by default).

(c) [Break]

You can configure the break function in this category.

Figure 2-18. [Break] Category**<1> [Type of breakpoints to be preferentially used]**

Specify from the following drop-down list the type of preferential breakpoint to be used with a single click of the mouse in the [Editor panel/Disassemble panel](#).
When setting a break point after the preferential break point type has been used up, the other break point type will be automatically selected.
See "[2.10.2 Stop the program at the arbitrary position \(breakpoint\)](#)" for details on breakpoints.

Software break	Sets software breakpoint preferentially.
Hardware break	Sets hardware breakpoint preferentially (default).

(d) [System]

You can configure the emulation system in this category.

For more information regarding the execution of a specified routine before the execution and after the break of a program, see "[2.9.4 Execute a specified routine \[E1\] \[E20\]](#)".

Figure 2-19. [System] Category

System	
Debug the program re-writing the on-chip PROGRAM ROM	No
Debug the program re-writing the on-chip DATA FLASH	No
Execute the specified routine immediately before execution of the user program	Yes
Routine to run immediately before execution starts	
Execute the specified routine immediately after the user program stops	Yes
Routine to run immediately after execution stops	

<1> [Debug the program re-writing the on-chip PROGRAM ROM]

Specify whether to debug programs that rewrite internal program ROM area, such as those that use ROM P/E mode.

Caution You cannot change this property while connected to E1.

<2> [Debug the program re-writing the on-chip DATA FLASH]

Specify whether to debug programs that rewrite internal data flash area, such as those that use data flash P/E mode.

Caution You cannot change this property while connected to E1.

<3> [Execute the specified routine immediately before execution of the user program]

Specify whether to execute a specified routine before executing the user program.

Caution You cannot change this property while the program is in execution.

<4> [Routine to run immediately before execution starts]

Specify the address to be executed immediately before the user program execution. This property is displayed only when [\[Execute the specified routine immediately before execution of the user program\]](#) property is set to [Yes].

Caution You cannot change this property while the program is in execution.

<5> [Execute the specified routine immediately after the user program stops]

Specify whether to execute a specified routine after the user program break.

Caution You cannot change this property while the program is in execution.

<6> [Routine to run immediately after execution stops]

Specify the address to be executed immediately after the user program break. This property is displayed only when [\[Execute the specified routine immediately after the user program stops\]](#) property is set to [Yes].

Caution You cannot change this property while the program is in execution.

(e) [Trace]

You can configure the trace function in this category.

Figure 2-20. [Trace] Category

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch

<1> [Usage of trace function]

[Trace] is displayed as the usage of trace function.
You cannot change the value of this property.

<2> [Operation after trace memory is full]

Select the trace acquisition mode from the following drop-down list.

Overwrite trace memory and continue execution	Continues overwriting the older trace data after the trace memory is full.
Stop trace	Stops writing the trace data after the trace memory is full.
Stop	Breaks after the trace memory is full.

<3> [Trace data type]

Select the type of data for which trace is to be acquired from the drop-down list.

The type of data that can be selected differs depending on the series of the microcontroller.

Following data types are displayed in the drop-down list.

- [RX600 series]
Branch, Branch+Data access, Data access
- [RX200 series]
Branch, Data access

(f) [Timer]

You can configure the timer function in this category.

Figure 2-21. [Timer] Category [RX600 Series]

Timer	
Use 64bit counter	No
Operating frequency[MHz]	25.0000

Figure 2-22. [Timer] Category [RX200 Series]

Timer	
Operating frequency[MHz]	25.0000

<1> [Use 64bit counter][RX600 series]

Specify whether to use two 32-bit counters or one 64-bit counter.

<2> [Operating frequency[MHz]]

Specify the operating frequency to be used when converting the count value to time.

Directly enter a number between 0.0001 and 999.999 (MHz) to specify the operating frequency.

When operating frequency is not specified, a count value will be displayed.

(3) [Download File Settings] tab

You can configure downloading to the debug tool in [\[Download File Settings\] tab](#). For the details of settings in each category, see ["2.5.1 Execute downloading"](#).

(4) [Hook Transaction Settings] tab

In the [\[Hook Transaction Settings\] tab](#), you can configure hook transaction for the debug tool.

See ["2.18 Setting Up the Hook Process"](#) for details on hook transaction and the settings in each category.

2.3.3 [E20]

Configure the operating environment on the [Property panel](#) below when using E20.

Note that the contents of the [Property panel](#) differ depending on the communication method (FINE communications [E20(Serial)] or JTAG communications [E20(JTAG)]) between E20 and the target system.

Figure 2-23. Property Panel [E20(Serial)]

Property	
RX E20(Serial) Property	
Internal ROM/RAM	
Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32
Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No
Connection with Emulator	
Emulator serial No.	E20: XXXXXXXXX
Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	FINE
FINE baud rate[bps]	2000000
Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX: 1000
Work RAM size[bytes]	1024
Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data
External Flash	
External flash definition file	[4]
Internal ROM/RAM	
Connect Settings Debug Tool Settings Download File Settings Hook Transaction Settings	

Figure 2-24. Property Panel [E20(JTAG)]

RX E20(JTAG) Property	
Internal ROM/RAM	
Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32
Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No
Connection with Emulator	
Emulator serial No.	E20:XXXXXXXXXX
Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	JTAG
JTAG clock[MHz]	16.5
Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024
Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data
External Flash	
External flash definition file	[4]
Internal ROM/RAM	

Connect Settings | Debug Tool Settings | Download File Settings | Hook Transaction Settings

Follow the steps below by selecting the corresponding tab on the [Property panel](#).

- (1) [\[Connect Settings\] tab](#)
- (2) [\[Debug Tool Settings\] tab](#)
- (3) [\[Download File Settings\] tab](#)
- (4) [\[Hook Transaction Settings\] tab](#)

(1) [\[Connect Settings\] tab](#)

In the [\[Connect Settings\] tab](#), you configure the connection with the debug tool for each one of the following categories.

- (a) [\[Internal ROM/RAM\]](#)
- (b) [\[Clock\]](#)
- (c) [\[Connection with Emulator\]](#)
- (d) [\[Connection with Target Board\]](#)
- (e) [\[Flash\]](#)
- (f) [\[Operating Modes of CPU\]](#)
- (g) [\[External Flash\]](#)

(a) [Internal ROM/RAM]

The configuration of internal ROM/RAM is displayed in this category.

Figure 2-25. [Internal ROM/RAM] Category

Internal ROM/RAM	
Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32

<1> [Size of internal ROM[KBytes]]

The internal ROM size to emulate is displayed (unit: Kbytes).

You cannot change the value of this property.

<2> [Size of internal RAM[Bytes]]

The internal RAM size to emulate is displayed (unit: bytes).

You cannot change the value of this property.

<3> [Size of DataFlash memory[KBytes]]

The data flash memory size is displayed (unit: Kbytes).

If the currently selected microcontroller does not incorporate the data flash, [0] is displayed.

You cannot change the value of this property.

(b) [Clock]

You can configure the clock in this category.

Caution You cannot change the property in this category while connected to E20.

Figure 2-26. [Clock] Category [HOCO]

Clock	
Main clock source	HOCO
Allow changing of the clock source on writing internal flash memory	No

Figure 2-27. [Clock] Category [EXTAL]

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Allow changing of the clock source on writing internal flash memory	No

<1> [Main clock source]

Select EXTAL frequency or internal HOCO as the main clock source. [EXTAL] will be displayed for microcontrollers with no internal HOCO.

<2> [Main clock frequency [MHz]]

Specify the main clock frequency (before multiplier).

Specify EXTAL frequency by directly entering a number between 0.0001 and 99.9999 (MHz). The entered value will be truncated to 5 decimal places. If the value is out of the specifiable range, it will be rounded to 0.0001 (when 0 or below) or to 99.9999 (when 100 or above).

This property is displayed only when you have selected [EXTAL] in the [\[Main clock source\]](#) property

<3> [Allow changing of the clock source on writing internal flash memory]

Specify whether to allow a debugger to operate the clock while the internal flash memory is being rewritten. When [Yes] is selected, any clock setting outside the guaranteed range of microcontroller operation will be modified by E20 to be in accord with the specified limits for reprogramming internal flash memory.

(c) [Connection with Emulator]

You can configure the connection between E20 and a host machine in this category.

Caution You cannot change the property in this category while connected to E20.

Figure 2-28. [Connection with Emulator] Category

Connection with Emulator	
Emulator serial No.	E20:XXXXXXXXXX

<1> [Emulator serial No.]

Serial numbers of all connected E20 emulators are displayed in the drop-down list.
Select the one to be connected to the target system.
The drop-down list is updated every time it is used.

(d) [Connection with Target Board]

You can configure the connection between E20 and the target board in this category.

Figure 2-29. [Connection with Target Board] Category [E20(Serial)]

Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	FINE
FINE baud rate[bps]	2000000

Figure 2-30. [Connection with Target Board] Category [E20(JTAG)]

Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	JTAG
JTAG clock[MHz]	16.5

<1> [Power target from the emulator. (MAX 200mA)]

[No] is displayed as the property value.
E20 does not support power supply function.

<2> [Communications method]

Displays the method of communication used by the E20 emulator for communicating with the microcontroller on the target system. Specifying [RX E20(Serial)] for a debug tool in the [Project Tree panel](#) will display [FINE] in this property, and specifying [RX E20(JTAG)] will display [JTAG].
You cannot change the value of this property.
For the details of debug tool selection, see "2.3.1 [Select the debug tool to use](#)".

<3> [JTAG clock[MHz]]

From the drop-down list, select the baud rate (JTAG clock) to be used by the E20 emulator for communicating with the microcontroller on the target system.

This property is displayed only when [JTAG] is selected in the [\[Communications method\]](#) property.

The following baud rate is displayed in the drop-down list.

- 16.5 (default), 12.38, 6.188, 3.094, 1.547

Cautions 1. This property cannot be changed while connected to E20.

2. Depending on the length or the method of JTAG signal wiring on the target system, it may not be possible to communicate using the selected JTAG clock. In such a case, reducing the JTAG clock may achieve successful communication.

<4> [FINE baud rate[bps]]

From the drop-down list, select the baud rate (FINE baud rate) to be used by the E20 emulator for communicating with the microcontroller on the target system.

This property is displayed only when [FINE] is selected in the [\[Communications method\]](#) property.

The following baud rate is displayed in the drop-down list.

- 2000000 (default), 750000, 500000, 250000

Cautions 1. This property cannot be changed while connected to E20.

2. Depending on the length or the method of FINE signal wiring on the target system, it may not be possible to communicate using the selected FINE baud rate. In such a case, reducing the FINE baud rate may achieve successful communication.

(e) [Flash]

You can configure the flash memory rewriting in this category.

Figure 2-31. [Flash] Category

Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024

<1> [Input Mode of ID code]

Specify the mode in which to input the ID codes.

Caution This property cannot be changed while connected to E20.

<2> [ID code]

Specify the ID code to use when reading the code from the memory.

If you have selected [Specify the ID code as a 32-digit hexadecimal] in the [\[Input Mode of ID code\]](#) property, enter the ID code in a 32-digit hexadecimal number. If you have selected [Specify the ID code as an ASCII code within 16 characters], enter the ID code using maximum 16 ASCII characters.

Cautions 1. This property cannot be changed while connected to E20.

2. If the ID code entered in ASCII characters is shorter than 16 characters, the unused space will be padded with 0.

<3> [Work RAM start address]

Specify the location address of the work RAM to be used by the debugger.

The size of work RAM to be used by the debugger firmware at a designated RAM location address is indicated in [Work RAM size[bytes]] property.

Cautions 1. This property cannot be changed while connected to E20.

2. This area can also be used by the user program since the memory contents are swapped and restored. However, you cannot designate this area as the source or destination of DMA or DTC function transfer.

<4> [Work RAM size[bytes]]

Displays the size of work RAM to be used by the debugger.

(f) [Operating Modes of CPU]

In this category, you configure the operating mode of the microcontroller to be emulated.

Caution This property cannot be changed while connected to E20.

Figure 2-32. [Operating Modes of CPU] Category

Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data

<1> [Mode pins setting]

Specify the operating mode set by the mode pin of the microcontroller.

<2> [Register setting]

Specify the operating mode to be set by the register.

The operating mode that can be specified depends on the type of the microcontroller.

<3> [Endian]

Displays the project endian. Acquires endian information from the project and displays its value. Can be selected only when the debug tool is disconnected.

(g) [External Flash]

In this category, you can configure external flash.

The settings in this category are required when downloading to an external flash memory. For more details regarding property setting, see “(5) Downloading files to external flash memory [E1] [E20]”.

Caution You cannot change the property in this category while connected to E20.

(2) [Debug Tool Settings] tab

In the [Debug Tool Settings] tab, you configure the basic settings of the debug tool for each one of the following categories.

(a) [Memory]

(b) [Access Memory While Running]

(c) [Break]

- (d) [System]
- (e) [Trace]
- (f) [Timer]

(a) [Memory]

You can configure the memory in this category.

Figure 2-33. [Memory] Category

Memory	
Memory mappings	[28]
[0]	On-chip RAM area
[1]	Reserved area
[2]	I/O registers area
[3]	Reserved area
[4]	I/O registers area
[5]	I/O registers area
[6]	Reserved area
[7]	Data flash area
[8]	Reserved area
[9]	Other memory area
[10]	Reserved area
[11]	I/O registers area
[12]	Reserved area
[13]	I/O registers area
[14]	Reserved area
[15]	External area (CS7)
[16]	External area (CS6)
[17]	External area (CS5)
[18]	External area (CS4)
[19]	External area (CS3)
[20]	External area (CS2)
[21]	External area (CS1)
[22]	Reserved area
[23]	Other memory area
[24]	Reserved area
[25]	Other memory area
[26]	Reserved area
[27]	On-chip ROM area
Verify on writing to memory	Yes

<1> [Memory mappings]

Current memory mapping status is displayed in detail for each type of memory area.

This property displays only the number of memory areas.

Expanding [Memory mapping] property will display the following sub-items.

- [Memory type]

Indicates the memory type of the corresponding area.

Each memory type corresponds to the following areas.

On-chip ROM area	Program ROM and data flash
On-chip RAM area	Internal RAM

On-chip ROM area	Program ROM and data flash
I/O registers area	Peripheral I/O register Divided into areas with different endians for display.
External area(CS7/CS6/.../CS0)	External address space CS0 to CS7 are displayed separately.
Reserved area	FCU-RAM, FCU firm, user boot
Other memory area	Areas other than those listed above

- [Start address]
Displays the start address of the corresponding area.
- [End address]
Displays the end address of the corresponding area.
- [Access width[bits]]
Displays the address width of the corresponding area.
When [Memory type] is an external area, the access width can only be changed when the debug tool is disconnected
- [Endian]
Displays the endians of the external area and the I/O register area.
When [Memory type] is an external area, the endian can only be changed when the debug tool is disconnected.

Caution Connecting to a debug tool (see "2.4.1 Connect the debug tool to CubeSuite+") will display details for each memory type.

<2> [Verify on writing to memory]

Specify whether to perform a verify check when the memory value is initialized from the drop-down list.
Select [Yes] to perform verification after download or when values are changed in the [Watch panel/](#)
[Memory panel](#).

(b) [Access Memory While Running]

You can configure the memory access while executing a program in this category.
The settings of this category are required when using the real-time display update function. See "(4) [Displaying and changing memory contents during program execution](#)" for details on the real-time display update function.

Figure 2-34. [Access Memory While Running] Category [E20(Serial)] [E20(JTAG)] [RX200 Series]]

Access Memory While Running	
Access by stopping execution	No
Update display during the execution	Yes
Display update interval[ms]	500

Figure 2-35. [Access Memory While Running] Category [E20(JTAG)] [RX600 Series]]

Access Memory While Running	
Access by stopping execution	No
Update the display during execution	Yes
Update interval[ms]	500
Enable the automatic update of realtime display	Yes

<1> [Access by stopping execution]

Specify from the drop-down list whether to allow access to the memory area while executing a program.
Select [Yes] to allow access ([No] is selected by default).

<2> [Update the display during execution]

Specify whether to update the display in the [Watch panel/Memory panel](#) while executing a program.
Select [Yes] to update the display (default).

<3> Update interval[ms]

This property is valid only when the [\[Update the display during execution\]](#) property is set to [Yes].
Specify the interval in 100ms unit to update the contents in the [Watch panel/Memory panel](#) display while executing a program.
Directly enter the Integer number between 100 and 65500 (rounding up the fractions less than 100ms) ([500] is specified by default).

<4> [Enable the automatic update of realtime display]

Specify whether to set RRM area automatically.
This property is displayed only when you have selected [Real-time RAM Monitor] in the [\[Usage of trace function\]](#) property. If you have selected [Trace] there, [No] is displayed instead.

Caution You cannot change this property while the program is in execution.

(c) [Break]

You can configure the break function in this category.

Figure 2-36. [Break] Category

**<1> [Type of breakpoints to be preferentially used]**

Specify from the following drop-down list the type of preferential breakpoint to be used with a single click of the mouse in the [Editor panel/Disassemble panel](#).
When setting a break point after the preferential break point type has been used up, the other break point type will be automatically selected.
See "[2.10.2 Stop the program at the arbitrary position \(breakpoint\)](#)" for details on breakpoints.

Software break	Sets software breakpoint preferentially.
Hardware break	Sets hardware breakpoint preferentially (default).

(d) [System]

You can configure the emulation system in this category.

For more information regarding the execution of a specified routine before the execution and after the break of a program, see "[2.9.4 Execute a specified routine \[E1\] \[E20\]](#)".

Figure 2-37. [System] Category

System	
Debug the program re-writing the on-chip PROGRAM ROM	No
Debug the program re-writing the on-chip DATA FLASH	No
Execute the specified routine immediately before execution of the user program	Yes
Routine to run immediately before execution starts	
Execute the specified routine immediately after the user program stops	Yes
Routine to run immediately after execution stops	

<1> [Debug the program re-writing the on-chip PROGRAM ROM]

Specify whether to debug programs that rewrite internal program ROM area, such as those that use ROM P/E mode.

Caution You cannot change this property while connected to E20.

<2> [Debug the program re-writing the on-chip DATA FLASH]

Specify whether to debug programs that rewrite internal data flash area, such as those that use data flash P/E mode.

Caution You cannot change this property while connected to E20.

<3> [Execute the specified routine immediately before execution of the user program]

Specify whether to execute a specified routine before executing the user program.

Caution You cannot change this property while the program is in execution.

<4> [Routine to run immediately before execution starts]

Specify the address to be executed immediately before the user program execution. This property is displayed only when [\[Execute the specified routine immediately before execution of the user program\]](#) property is set to [Yes].

Caution You cannot change this property while the program is in execution.

<5> [Execute the specified routine immediately after the user program stops]

Specify whether to execute a specified routine after the user program break.

Caution You cannot change this property while the program is in execution.

<6> [Routine to run immediately after execution stops]

Specify the address to be executed immediately after the user program break. This property is displayed only when [\[Execute the specified routine immediately after the user program stops\]](#) property is set to [Yes]

Caution You cannot change this property while the program is in execution.

(e) [Trace]

You can configure the trace function in this category.

Figure 2-38. [Trace] Category [E20(Serial) [RX600 Series]]

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch
External trace output	CPU execution
Trace memory size[MByte]	1

<1> [Usage of trace function]

Specify whether to use it as Real-time RAM Monitor (RRM) utilizing the trace function. If you select [Real-time RAM Monitor], part of the trace functions will be disabled.

For details on applicable restrictions, see "(4) [Displaying and changing memory contents during program execution](#)".

This property can be changed only when the program is not running.

Caution [E20(Serial)] [E20(JTAG) [RX200 Series]]

Real-time RAM Monitor function is not supported. As such, the property value is displayed as [Trace] and becomes unchangeable.

<2> [Operation after trace memory is full]

Select the trace acquisition mode from the following drop-down list.

Overwrite trace memory and continue execution	Continues overwriting the older trace data after the trace memory is full.
Stop trace	Stops writing the trace data after the trace memory is full.
Stop	Breaks after the trace memory is full.

Caution [E20(JTAG) [RX600 Series]]

If you have selected [Real-time RAM Monitor] in [\[Usage of trace function\]](#) property, the property value is displayed as [Overwrite trace memory and continue execution] and becomes unchangeable.

<3> [Trace data type]

Select the type of data for which trace is to be acquired from the drop-down list.

The type of data that can be selected differs depending on the series of the microcontroller.

Following data types are displayed in the drop-down list.

- [RX600 series]
Branch, Branch+Data access, Data access
- [RX200 series]
Branch, Data access

Caution [E20(JTAG) [RX600 Series]]

If you have selected [Real-time RAM Monitor] in [\[Usage of trace function\]](#) property, the property value is displayed as [Data access] and becomes unchangeable.

<4> [External trace output][E20(JTAG)]

From the following drop-down list, select the method in which to externally output the trace acquisition data.

CPU execution	CPU execution given priority over trace output. Trace information may be lost if output.
Trace output	Trace output given priority over CPU execution. CPU execution stops during trace output, affecting real-time performance.
Do not output	Only the internal buffer of the microcontroller will be used, with no output of trace information.

Caution Caution: If you have selected [Real-time RAM Monitor] in [Usage of trace function] property, [Do not output] cannot be selected from the drop-down list.

<5> [Trace memory size[MByte]][E20(JTAG)]

Specify the size of memory used to retain the trace data.

The following memory sizes are displayed in the drop-down list.

- 1 (default), 2, 4, 8, 16, 32

Caution If you have selected [Real-time RAM Monitor] in [Usage of trace function] property, the property value is displayed as [1] and becomes unchangeable.

(f) [Timer]

You can configure the timer function in this category.

Figure 2-39. [Timer] Category [RX600 Series]

Timer	
Use 64bit counter	No
Operating frequency[MHz]	25.0000

Figure 2-40. [Timer] Category [RX200 Series]

Timer	
Operating frequency[MHz]	25.0000

<1> [Use 64bit counter][RX600 series]

Specify whether to use two 32-bit counters or one 64-bit counter.

<2> [Operating frequency[MHz]]

Specify the operating frequency to be used when converting the count value to time.

Directly enter a number between 0.0001 and 999.999 (MHz) to specify the operating frequency.

When operating frequency is not specified, a count value will be displayed.

(3) [Download File Settings] tab

You can configure downloading to the debug tool in [Download File Settings] tab.

For the details of settings in each category, see "2.5.1 Execute downloading".

(4) [Hook Transaction Settings] tab

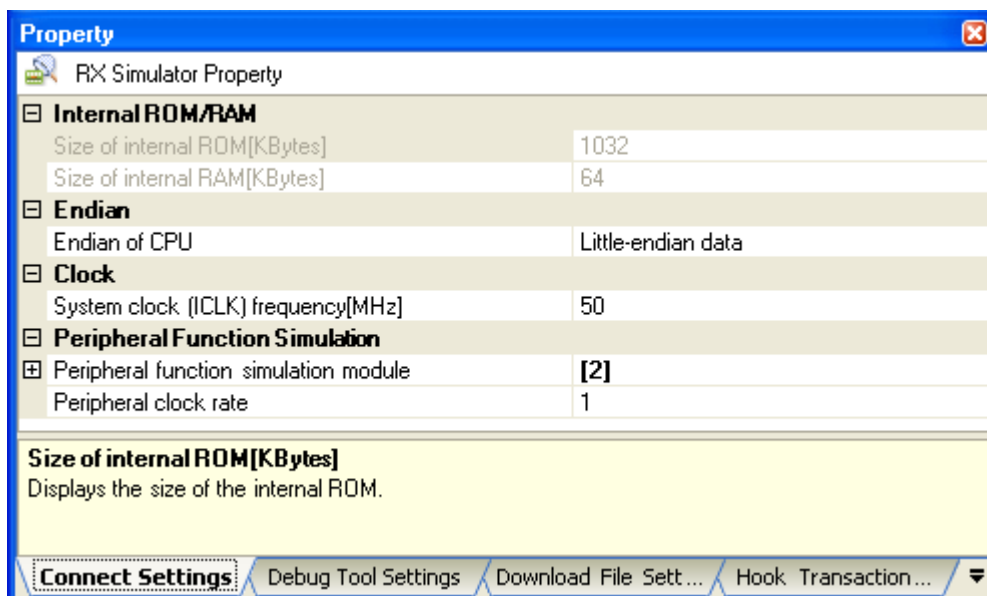
In the [Hook Transaction Settings] tab, you can configure hook transaction for the debug tool.

See "2.18 Setting Up the Hook Process" for details on hook transaction and the settings in each category.

2.3.4 [Simulator]

Configure the operating environment on the [Property panel](#) below when using Simulator.

Figure 2-41. Operating Environment Settings [Simulator] (Property Panel)



Follow the steps below by selecting the corresponding tab on the [Property panel](#).

- (1) [\[Connect Settings\] tab](#)
- (2) [\[Debug Tool Settings\] tab](#)
- (3) [\[Download File Settings\] tab](#)
- (4) [\[Hook Transaction Settings\] tab](#)

(1) **[Connect Settings] tab**

In the [\[Connect Settings\] tab](#), you can configure the connection with the debug tool for each one of the following categories.

- (a) [\[Internal ROM/RAM\]](#)
- (b) [\[Endian\]](#)
- (c) [\[Clock\]](#)
- (d) [\[Peripheral Function Simulation\]](#)

(a) **[Internal ROM/RAM]**

The size of internal ROM/RAM of the selected microcontroller is displayed in this category.

Figure 2-42. [Internal ROM/RAM] Category [Simulator]

Internal ROM/RAM	
Size of internal ROM[KBytes]	1032
Size of internal RAM[KBytes]	64

<1> **[Size of internal ROM[KBytes]]**

The internal ROM size to simulate is displayed (unit: Kbytes).

You cannot change the value of this property.

<2> [Size of internal RAM[KBytes]]

The internal RAM size to simulate is displayed (unit: Kbytes).

You cannot change the value of this property.

(b) [Endian]

You can configure endian in this category.

Figure 2-43. [Endian] Category [Simulator]

Endian	
Endian of CPU	Little-endian data

<1> [CPU Endian]

Specify the endian for the CPU.

By default, the endian selected in the build tool property is displayed.

Caution You cannot change this property while connected to the Simulator.

(c) [Clock]

You can configure clock in this category.

Figure 2-44. [Clock] Category [Simulator]

Clock	
System clock (ICLK) frequency[MHz]	50

<1> [System clock (ICLK) frequency [MHz]]

Specify the clock frequency for the CPU (unit: MHz).

Directly enter the value between 1 [MHz] and 1.000[MHz] ([100] is specified for the RX600 Series and [50] for the RX200 Series by default).

Caution You cannot change this property while connected to the Simulator.

(d) [Peripheral Function Simulation]

You can configure simulation of peripheral functions in this category.

Figure 2-45. [Peripheral Function Simulation] Category

Peripheral Function Simulation	
Peripheral function simulation module	[2]
[0]	CMT
[1]	ICU
Peripheral clock rate	1

<1> [Peripheral function simulation module]

It displays the name of available peripheral function simulation modules and lets you select from the drop-down list whether to use each module being displayed.

Select [Use] when using the module ([Not Use] is selected by default).

You cannot change the name of peripheral function simulation modules on this panel.

<2> [Peripheral clock rate]

Specify from the drop-down list the peripheral-to-internal clock ratio that shows the number of internal clock that is equivalent to 1 peripheral clock.

The following clock rates are displayed in the drop-down list.

1(Default), 2, 3, 4, 6, 8, 12, 16, 24, 32, 64

Caution You cannot change this property while connected to the Simulator.

(2) [Debug Tool Settings] tab

In the [\[Debug Tool Settings\] tab](#), you can configure the debug tool for each one of the following categories.

- (a) [\[Memory\]](#)
- (b) [\[Access Memory While Running\]](#)
- (c) [\[Trace\]](#)
- (d) [\[Coverage\]](#)
- (e) [\[Stream input/output\]](#)
- (f) [\[Execution mode\]](#)
- (g) [\[Instruction decode cache\]](#)

(a) [Memory]

You can configure the memory in this category.

Figure 2-46. [Memory] Category [Simulator]

Memory	
Memory mappings	
	[19]
⊕ [0]	Internal RAM area
⊕ [1]	Non-map area
⊕ [2]	I/O registers area
⊕ [3]	Non-map area
⊕ [4]	I/O registers area
⊕ [5]	I/O registers area
⊕ [6]	Non-map area
⊕ [7]	I/O registers area
⊕ [8]	I/O registers area
⊕ [9]	I/O registers area
⊕ [10]	Internal ROM area
⊕ [11]	Non-map area
⊕ [12]	I/O registers area
⊕ [13]	Non-map area
⊕ [14]	I/O registers area
⊕ [15]	Non-map area
⊕ [16]	Internal ROM area
⊕ [17]	Non-map area
⊕ [18]	Internal ROM area

<1> [Memory mappings]

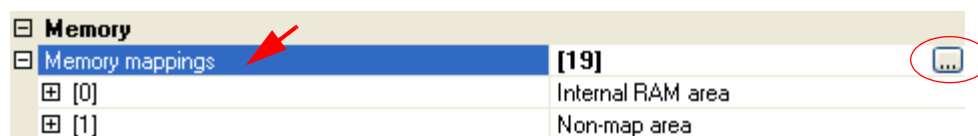
Current memory mapping status is displayed for each type of memory area.

The memory mapping status cannot be changed on this panel. When necessary, you can add a memory mapping in the [Memory Mapping dialog box \[Simulator\]](#). This dialog box can be displayed by clicking on

the [...] button which appears on the right end of the setting field after selecting [Memory Mapping] property.

See the section for the [Memory Mapping dialog box \[Simulator\]](#) for details on how to configure the parameters.

Figure 2-47. Opening the Memory Mapping Dialog Box



(b) [Access Memory While Running]

You can configure the memory access while executing a program in this category.

The settings of this category are required when using the real-time display update function. See "(4) [Displaying and changing memory contents during program execution](#)" for details on the real-time display update function.

Figure 2-48. [Access Memory While Running] Category [Simulator]

Access Memory While Running	
Update display during the execution	Yes
Display update interval[ms]	500

<1> [Update display during the execution]

Specify from the drop-down list whether to update the display in the [Watch panel/Memory panel](#) during a program execution.

Select [Yes] to update the display (default).

<2> [Display update interval[ms]]

This property is valid only when the [\[Update display during the execution\]](#) property is set to [Yes].

Specify the interval in 100ms unit to update the contents in the [Watch panel/Memory panel](#) display while executing a program.

Directly enter the Integer number between 100 and 65500 (rounding up the fractions less than 100ms) ([500] is selected by default).

(c) [Trace]

You can configure the trace function in this category.

See "[2.13 Collecting an Execution History](#)" for details on the trace function and this category configuration.

(d) [Coverage]

You can configure the coverage function in this category.

See "[2.15 Measure Coverage \[Simulator\]](#)" for details on the coverage function and this category configuration.

(e) [Stream input/output]

You can configure the stream input/output for performing standard input/output or file input/output to/from the user program in this category.

Figure 2-49. [Stream input/output] Category

Stream I/O	
Use stream I/O function	No
Stream I/O address	HEX 0

<1> [Use stream input/output function]

Specify from the drop-down list whether to use the stream input/output function.

Select [Yes] when using this function ([No] is selected by default).

<2> [Stream input/output address]

Specify the address at which to start the stream input/output.

Directly enter the desired address ([0x00000000] is specified by default).

(f) [Execution mode]

You can configure the operation that is required in the event of simulation error or exception in this category.

Figure 2-50. [Execution mode] Category

Execution Mode	
Select execution mode	Stop
Stop when undefined instruction exception occurs	Yes
Stop when privileged instruction exception occurs	Yes
Stop when access exception occurs	Yes
Stop when floating-point exception occurs	Yes
Stop when interrupt occurs	Yes
Stop INT instruction is executed	Yes
Stop BRK instruction is executed	Yes

<1> [Select execution mode]

In this property, you can select the execution mode from the following drop-down list.

Stop	Stops simulation (default).
Continue	Continues simulation (Property items in the bottom section will be enabled.)

<2> [Stop when undefined instruction exception is encountered]

This property is displayed only when you have selected [Stop] in [\[Select execution mode\]](#) property.

Specify from the drop-down list whether to stop when undefined instruction exception is encountered.

Select [Yes] if you wish to stop at the occurrence of undefined instruction exception (default).

<3> [Stop when privileged instruction exception is encountered]

This property is displayed only when you have selected [Stop] in [\[Select execution mode\]](#) property.

Specify from the drop-down list whether to stop when privileged instruction exception is encountered.

Select [Yes] if you wish to stop at the occurrence of privileged instruction exception (default).

<4> [Stop when access exception is encountered]

This property is displayed only when an MPU module exists in [\[Peripheral function simulation module\]](#)

property and you have selected [Stop] in [\[Select execution mode\]](#) property. Specify from the drop-down

list whether to stop when access exception is encountered. Select [Yes] if you wish to stop at the

occurrence of access exception (default).

<5> [Stop when floating-point exception is encountered][RX600 Series]

This property is displayed only when you have selected [Stop] in [\[Select execution mode\]](#) property. Specify from the drop-down list whether to stop when floating-point exception is encountered. Select [Yes] if you wish to stop at the occurrence of floating-point exception (default).

<6> [Stop when interrupt is encountered]

This property is displayed only when you have selected [Stop] in [\[Select execution mode\]](#) property. Specify from the drop-down list whether to stop when interrupt is encountered. Select [Yes] if you wish to stop at the occurrence of interrupt (default).

<7> [Stop INT instruction is executed]

This property is displayed only when you have selected [Stop] in [\[Select execution mode\]](#) property. Specify from the drop-down list whether to stop when INT instruction is executed. Select [Yes] if you wish to stop at the execution of INT instruction (default).

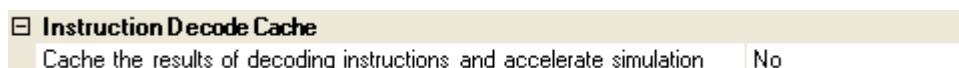
<8> [Stop BRK instruction is executed]

This property is displayed only when you have selected [Stop] in [\[Select execution mode\]](#) property. Specify from the drop-down list whether to stop when BRK instruction is executed. Select [Yes] if you wish to stop at the execution of BRK instruction (default).

(g) [Instruction decode cache]

In this category, you can configure instruction decode cache, a function which retains decode result during instruction execution so that it can be used again when the instruction at the same address is executed.

Figure 2-51. [Instruction decode cache] Category

**<1> [Increase simulation speed by caching instruction decode result]**

Specify from the drop-down list whether to enable instruction decode cache function. Select [Yes] to enable this function ([No] is selected by default).

Caution As the instruction decode cache function reuses the decode result, it cannot be used in programs that employ self-modifying code. If an instruction is modified as a result of unintended program operation, error detection may not be performed properly.

(3) [Download File Settings] tab

In the [\[Download File Settings\] tab](#), you can configure download setting for the debug tool. See ["2.5.1 Execute downloading"](#) for details on each category configuration.

(4) [Hook Transaction Settings] tab

In the [\[Hook Transaction Settings\] tab](#), you can configure hook transaction for the debug tool. See ["2.18 Setting Up the Hook Process"](#) for details on each category configuration and hook transaction.

2.4 Connect to/Disconnect from the Debug Tool

This section describes connection to and disconnection from the debug tool including hot plug-in connection.

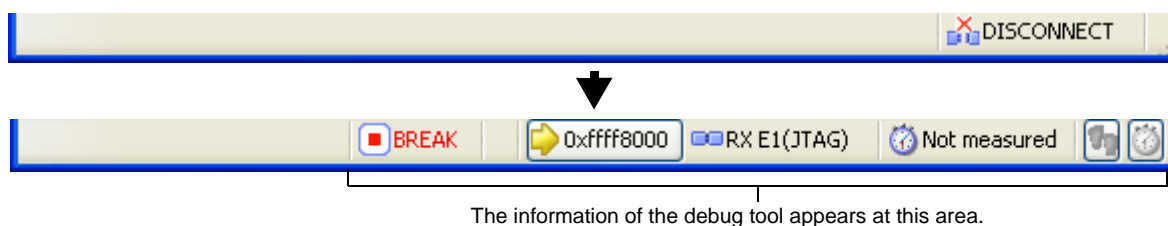
2.4.1 Connect the debug tool to CubeSuite+


Select the [Debug] menu >> [Connect to Debug Tool] to connect to the debug tool selected in the currently active project.


After succeeding in the connection to the debug tool, the **Status bar** of the **Main window** changes as follows:

For details on each item displayed on the **Status bar**, see the section of the "Main window".


Figure 2-52. Statusbar Indicating the Successful Connection to the Debug Tool



Remark When the  button on the **Debug toolbar** is clicked, it will download the specified file after connecting to the debug tool (see "2.5.1 Execute downloading").

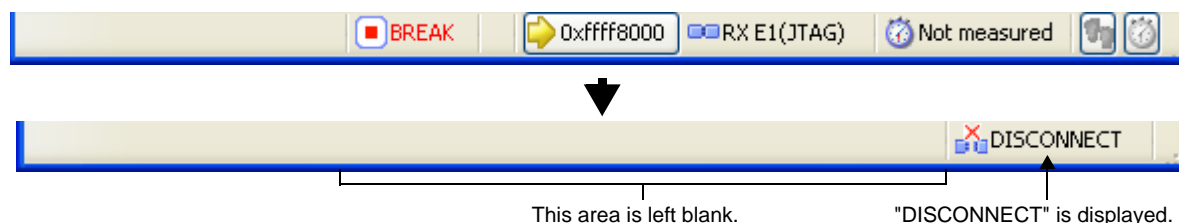
When the  button on this toolbar is clicked, it will build the project, connect to the debug tool and then download the specified file.

2.4.2 Disconnect the debug tool from CubeSuite+

Click the  button on the **Debug toolbar** to cut off the communication with the connected debug tool.

After disconnecting from the debug tool, the **Status bar** on the **Main window** changes as follows:

Figure 2-53. Statusbar Indicating the Disconnection from the Debug Tool



Caution The debug tool cannot be disconnected from CubeSuite+ while the program is running. If you wish to disconnect the debug tool, stop the program in advance.

Remark Disconnecting the debug tool will close all the panels and dialog boxes that can be displayed only during the connection.

2.4.3 Connect the debug tool to CubeSuite+ using hot plug-in [E1(JTAG)] [E20(JTAG)]

With hot plug-in function, you can connect the debug tool to the CubeSuite+ and debug the program while it is in execution.

Follow the steps below to establish hot plug-in connection.

(1) Execute the program

Execute the program which has been downloaded onto the microcontroller on the target system without connecting to the emulator.

(2) Specify the debug tool

In the active project, specify the debug tool which supports hot-plug in connection ([E1(JTAG)]/[E20(JTAG)]).

Remark [E1(Serial)] and [E20(Serial)] do not support hot plug-in connection.

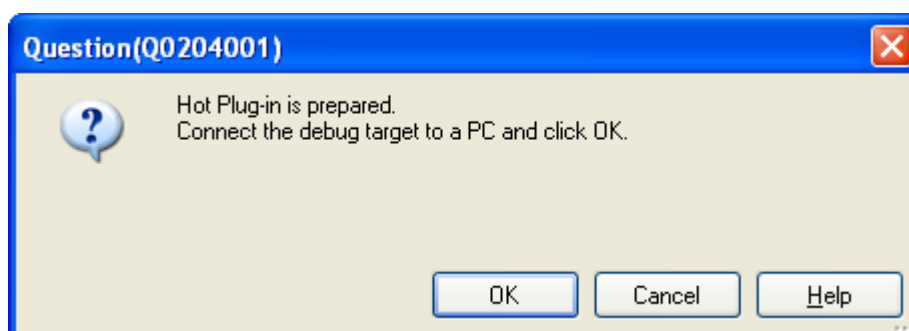
(3) Connect the debug tool to CubeSuite+ using hot plug-in

Select [Hot Plug-in] from [Debug] menu to initiate the preparation for hot plug-in connection.

(4) Connect to the target system

Following message will appear once you are ready to start hot plug-in connection. Connect the emulator to the target system and click [OK]. This will start the communication with the debug tool which is selected in the currently active project.

Figure 2-54. Message Indicating that Hot Plug-in Connection Is Ready to be Started



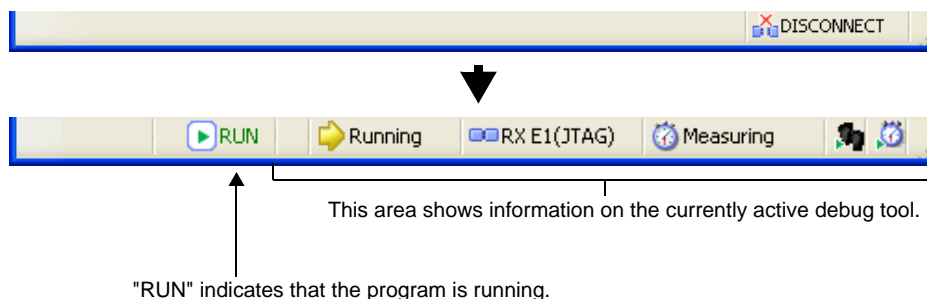
Caution [E1(JTAG)]

To establish hot plug-in connection, you need to connect the emulator to the target system via "Hot-plug Adapter" (R0E000010ACB00) which is optionally available.

(5) Hot plug-in connection completed

Once the connection to the debug tool is successfully completed, the **Status bar** on the **Main window** will change as shown below. For details on each item displayed on the **Status bar**, see the section of the "Main window".

Figure 2-55. Statusbar Indicating the Successful Hot plug-in Connection to the Debug Tool



- Cautions 1.** Trace function and Real-time display updating function will not be available until the program breaks following the completion of hot plug-in connection.
2. The realtime RAM monitor function (RRM function) will not be available until the program breaks following the completion of hot plug-in connection. Do not select [Real-time RAM monitor] for the [Usage of trace function] property under the [Trace] category.
 3. Once the hot plug-in connection is established, all the events that have been set in the project as the user information (excluding built-in event) will be deleted.
 4. When conducting hot plug-in connection, do not use a project for which software break point is set as it may result in unsuccessful connection. Check that all the software break points are deleted in the **Events panel** of the active project before starting hot plug-in connection (see [2.17.5 Deleting events](#)).
 5. The emulator stops the program temporarily for approximately 800μs to check the ID code at hot plug-in connection (CPU clock: 100 MHz, JTAG clock: 16.5 MHz).
 6. [RX630, RX631, RX63N group]
Confirm that the Endian value of Endian Select Register (MDEB, MDES) written on the microcontroller matches the Endian information of the project. See the hardware manual of the microcontroller for the details of Endian Select Register.

2.5 Download and Upload

This section describes how to download the program to be debugged (e.g., load module files) into CubeSuite+, and how to upload the memory content under debug from CubeSuite+ to files.

2.5.1 Execute downloading

Download the load module files to be debugged.

Following the procedure below, make the necessary settings for downloads on the [Property panel's \[Download File Settings\] tab](#) before you execute a download.

(1) Setting the [Download] category

Figure 2-56. [Download] Category [E1(JTAG)][E1(Serial)][E20(JTAG)][E20(Serial)]

<input type="checkbox"/> Download	
<input type="checkbox"/> Download files	[1]
<input type="checkbox"/> [0]	DefaultBuild\test_RX.abs
File	DefaultBuild\test_RX.abs
File type	Load module file
Download object	Yes
Download symbol information	Yes
Specify the PIC/PID offset	No
CPU Reset after download	Yes
Erase flash ROM before download	No
Automatic change method of event setting position	Suspend event

Figure 2-57. [Download] Category [Simulator]

<input type="checkbox"/> Download	
<input type="checkbox"/> Download files	[1]
<input type="checkbox"/> [0]	DefaultBuild\test_RX.abs
File	DefaultBuild\test_RX.abs
File type	Load module file
Download object	Yes
Download symbol information	Yes
Specify the PIC/PID offset	No
CPU Reset after download	Yes
Automatic change method of event setting position	Suspend event

(a) [Download files]

Displays the file names to be downloaded, as well as download conditions (the numeric value in property value "[]" indicating the number of files currently specified to be downloaded).

The files to be downloaded here are those files that have been specified as the subject to build in the main project and sub-projects. This means that the same files that were specified in the project are automatically determined^{Note} to be the download files.

However, the files to be downloaded and the download conditions can be manually changed. In this case, see ["2.5.2 An applied method of download"](#).

Note To download the load module files created by an external build tool (e.g., compilers and assemblers other than the build tools supplied with CubeSuite+), a debug-only project needs to be created. If you use a debug-only project as the subject to debug, add your download files to the download file node on project tree. The files to be downloaded will be reflected in this property.

For details on how to use "external build tools" and details about "debug-only projects," see the separate edition, "CubeSuite+ Start".

(b) [CPU Reset after download]

Specify from the drop-down list whether or not to reset the CPU after a download is completed.

To reset the CPU, select [Yes] (default).

(c) [Erase flash ROM before download]

Specify from the drop-down list whether or not to erase the flash ROM before a download is executed.

To erase the flash ROM, select [Yes]. (By default, [No] is selected).

(d) [Automatic change method of event setting position]

If, as debug work proceeds, a program which has had changes added is downloaded again, the position (address) at which a currently set event is set may happen to be in the middle of an instruction. In such a case, use this property to specify how the subject event should be handled.

Select one of choices from the drop-down list below.

Move to the head of instruction	Resets the subject event at the beginning address of the instruction.
Suspend event	Leaves the subject event pending (default).

However, specification on this property applies to only the event set positions for which debug information is nonexistent. For the event set positions that have debug information, the event is always moved to the beginning of the source text line.

(2) Setting the [Debug Information] category

Figure 2-58. [Debug information] Category

☐ Debug Information	
Execute to the specified symbol after CPU Reset	Yes
Specified symbol	_main
Specify the debugged overlay section	No

(a) [Execute to the specified symbol after CPU Reset]

Specify from the drop-down list whether or not to run the program up to a specified symbol position after the CPU is reset, or after a download is completed (only when you specified [Yes] on the [\[CPU Reset after download\]](#) property).

To run the program up to a specified symbol position, select [Yes] (default).

Remark If, while you specified [Yes] on the [\[CPU Reset after download\]](#) property, you specify [Yes] on this property, the [Editor panel](#) opens automatically after a download is completed, with the source text at the position you specified on [\[Specified symbol\]](#) property displayed on it.
Also, if you specify [No] here, this panel opens, with the reset address displayed on it. (If the reset address has no source text mapped to it, the relevant place is displayed on the [Disassemble panel](#).)

(b) [Specified symbol]

This property is displayed only when you specified [Yes] on the [\[Execute to the specified symbol after CPU Reset\]](#) property.

Specify a position at which you want the program being run after the CPU is reset to be halted.
 Enter an address expression directly in the range 0 to "end address of address space" to specify the position.
 (By default, [_main] is specified.)
 However, if the specified address expression cannot be converted into an address, the program is not executed.

Remark Normally, specify the following.

For assembler source:	Beginning label equivalent to the main function
For C source:	Symbol given at the beginning of the main function name

(c) [Specify the debugged overlay section]

"Yes" is displayed when overlay sections (see "2.8 Setting Overlay Sections" for details) exist in the load module.

Otherwise "No" is displayed. This entry cannot be changed.


(d) [Overlay sections]

Address groups where overlay sections exist are displayed.

This property only appears when "Yes" is displayed on the [Specify the debugged overlay section] property.

Caution By default, CPU reset automatically occurs after downloading the file, and then the program is executed to the specified symbol position. If this operation above is not needed, specify [No] with both of the [CPU Reset after download] and [Execute to the specified symbol after CPU Reset] property.

(3) Executing a download

Click the  button in the Debug toolbar.

If this operation is performed while CubeSuite+ is disconnected from the debug tool, it is automatically connected to the debug tool before a download is executed.

Remark To download a file which has had changes added in the course of debug work again, choose [Build & Download to Debug Tool] from the [Debug] menu on the Main window, which will help you build and download easily.

The load module files have been successfully download, the Editor panel opens automatically, with the source text of the downloaded files displayed on it.

Remark A process can be set that automatically rewrites I/O register or CPU register values with specified values before or after a download (see Section "2.18 Setting Up the Hook Process").

2.5.2 An applied method of download

The files to download, as well as the download conditions applied at download time can be changed.

CubeSuite+ permits files in the following formats to be downloaded.

Table 2-2. Downloadable File Formats

File Format	Extension	Remark
Load module format	abs	
	<i>Extension</i> [IAR] ^{Note}	Usable exclusively for debug-only projects
Intel Hex format - Standard (offset address) - 02 (segment address code) - 04 (extension linear address code)	hex	03 (start segment address code) and 05 (start linear address code) are ignored.
Motorola S type Hex format - (S0, S1, S9-16 bits) - (S0, S2, S8-24 bits) - (S0, S3, S7-32 bits)	mot	
Binary data format	bin	

Note Precautions to take when using IAR compilers (made by IAR Systems of Sweden)

- Supported versions
 - IAR Embedded Workbench for Renesas (Ver. 2.30.1 or above)
- Supported options
 - General options: Byte order:
 - Little endian/ big endian
 - double type size: 32 bits/ 64 bits
 - int type size: 16 bits/ 32 bits
 - Denormal number: Handled as zero
 - Data model: Far
 - Position-dependence: Off
 - C/C++ compilers:
 - Language: C * C++ and GNU C extended specification are not supported.
 - Derived languages of C: C89/ C99
 - (LVA permission, Off; C++ inline semantic, Off)
 - Type of CHAR: Signed/ unsigned
 - Optimization: None/ low
 - Output: Generate debug information, On
 - Assemblers:
 - Language: Discriminate user symbols between uppercase and lowercase, On
 - Output: Generate debug information, On
 - Linkers:
 - Output: Include debug information in output files, On
- Precautions to take when debugging
 - Static variables in file cannot be referenced on [Watch panel](#), etc. The static variables defined in functions can be referenced.
 - It is only the current function that can be displayed on the [Call Stack panel](#).
 - If a defined variable is composed of multiple registers, the value of the subject variable cannot be displayed correctly.
 - Example 1: long long data = 0x123456789abcdef0
 - // A case where the variable "data" is mapped to the registers R7 and R8
 - Example 2: struct aaa { char a; short b; long c; char d;};

```
struct aaa sA = { 'S', 0x4154, 0x4E455452, 'D'};
```

```
// A case where the variable "sA" is mapped to the registers R7 and R8
```

- If a variable is deleted as a result of optimization by the compiler, the subject variable cannot be referred to in the debugger.
- If a variable is temporarily mapped to a register as a result of optimization by the compiler, the value of the variable may not be displayed correctly on the [Watch panel](#), etc.

To change the download files, or set the download conditions applied at download time, use the [Download Files dialog box](#) shown below.

The Download Files dialog box is opened by clicking the [...] button that is displayed at the right edge in the column of the [\[Download files\]](#) property when you select it in the [\[Download\]](#) category on the [Property panel](#)'s [\[Download File Settings\]](#) tab.

Figure 2-59. Opening the Download Files Dialog Box

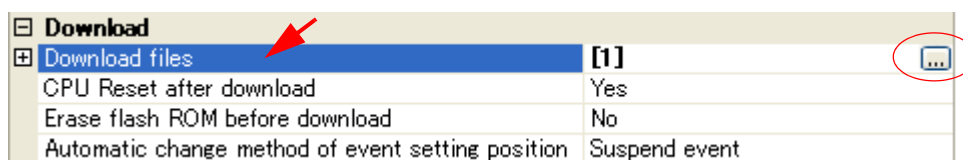
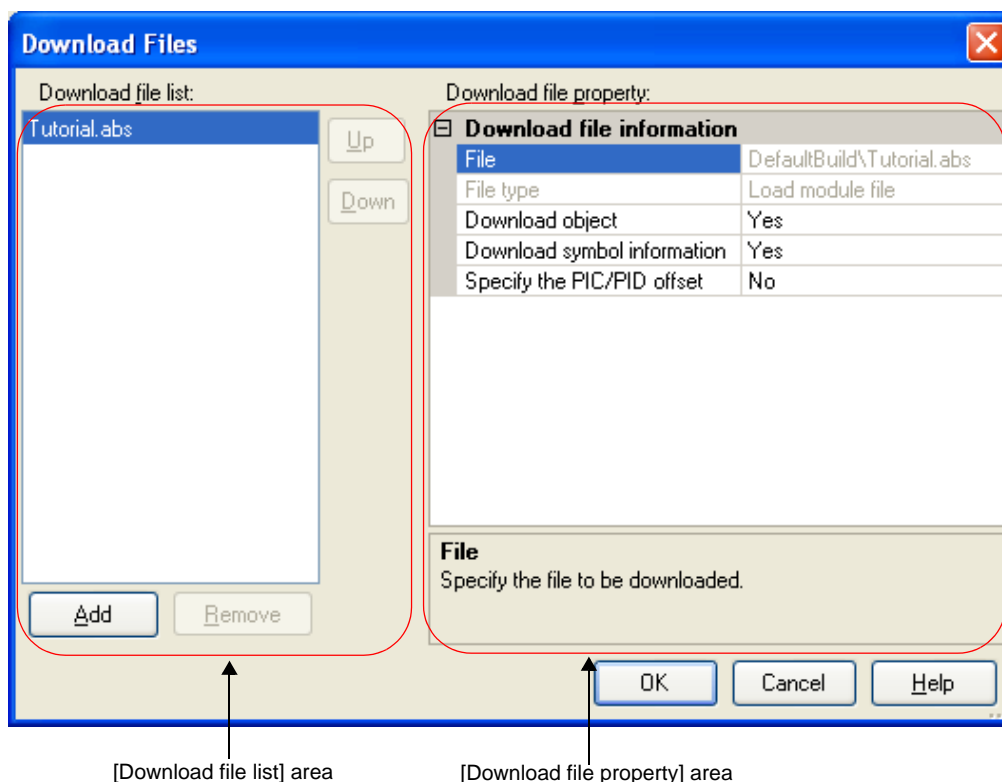


Figure 2-60. An Applied Method of Download (Download Files Dialog Box)



Here, a method on how to set in the [Download Files dialog box](#) shown above is described, using the following cases as examples.

- (1) [Changing the download conditions for load module files](#)
- (2) [Adding a download file \(*.hex, *.mot, or *.bin\)](#)

- (3) Downloading multiple load module files
- (4) Performing source-level debug with hex format, Motorola S format, or binary data format files at the source level
- (5) Downloading files to external flash memory [E1] [E20]

(1) Changing the download conditions for load module files

To change the download conditions for load module files (whether to load object information or symbol information) under which they are downloaded, follow the procedure described below as you fill in the [Download Files dialog box](#).

(a) Selection of a load module file

In the [\[Download File List\] area](#), select the load module file to download.

(b) Alteration of download conditions

In the [\[Download File List\] area](#), you'll see the download conditions for the currently selected load module file displayed in it.

For each displayed item, change their setting.

Download object	Specify whether or not to download object information from a specified file.		
	Default	Yes	
	Method to change	By selecting from drop-down list	
	Specifiable value	Yes	Downloads object information.
		No	Does not download object information.
Download symbol information	Specify whether or not to download symbol information from a specified file ^{Note 1} .		
	Default	Yes	
	Method to change	By selecting from drop-down list	
	Specifiable value	Yes	Downloads symbol information.
		No	Does not download symbol information.
Specify the PIC/PID offset	Specify whether or not to change the positions of PIC (Position Independent Code) and PID (Position Independent Data) areas of the load module to be downloaded from those at which they were when the load module was created. When this item is changed to [Yes], "PIC Offset" and "PID Offset" are displayed in the sub-item.		
	Default	No	
	Method to change	By selecting from drop-down list	
	Specifiable value	Yes	Specifies PIC/PID offset ^{Note 2} .
		No	Does not specify PIC/PID offset.

PIC Offset	Enter an offset value from an address that was in the load module when it was created. For example, if the program section's start address begins with the address 0x1000, enter "1000" for this item. The subject section is downloaded to the address 0x2000.	
	Default	0
	Method to change	By entering directly from the keyboard
	Specifiable value	Hexadecimal number in the range 0x0 to 0xFFFFFFFF
PID Offset	Enter an offset value to be set in the PID register that was specified in the load module when it was created. For example, to set 0x200 in the PID register when the load module is executed, enter "200" for this item.	
	Default	0
	Method to change	By entering directly from the keyboard
	Specifiable value	Hexadecimal number in the range 0x0 to 0xFFFFFFFF

- Notes**
1. Unless symbol information is downloaded, source-level debugging cannot be performed.
 2. If you select [Yes] for load modules created not using the PIC/PID functions (see Section "2.7 Usage of PIC/PID Function"), debug operation cannot be guaranteed.

(c) Clicking the [OK] button

Settings you've made in this dialog box are enabled, with the download conditions changed.

(2) Adding a download file (*.hex, *.mot, or *.bin)

To add a file to download (hex format (*.hex), Motorola S format (*.mot), or binary data format (*.bin)), follow the procedure described below as you fill in the [Download Files dialog box](#).

(a) Clicking the [Add] button

When you click the [Add] button, a blank item ("-") is displayed on the last line in the [\[Download File List\] area](#).

(b) Setting the properties of a download file to add

In the [\[Download file property\] area](#), select a download file you want to add and set its download conditions. For each displayed item, make the following setting.

When setting is completed, the file name you've specified here is reflected in the blank item in the [\[Download File List\] area](#).

File	Specify a download file to add (hex format (*.hex), Motorola S format (*.mot), or binary data format (*.bin)) (specifiable in up to 259 characters).	
	Default	Blank
	Method to change	By entering directly from the keyboard, or specifying in the Select Download File dialog box that is opened by clicking the [...] button displayed at the right edge of this property when it is selected
	Specifiable values	See "Table 2-2. Downloadable File Formats".

File type	Specify the file type of a download file to add. Here, select [Hex File], [S Record File] or [Binary Data File].	
	Default	Load module file
	Method to change	By selecting from drop-down list
	Specifiable values	Either one of the following - Load module file - Hex file - S record file - Binary data file
Offset	This item is displayed only when the file to download is a hex format (*.hex) or Motorola S format (*.mot). Specify an offset value from the address at which a download of a specified file begins.	
	Default	0
	Method to change	By entering directly from the keyboard
	Specifiable values	Hexadecimal number in the range 0x0 to 0xFFFFFFFF
Start address	This item is displayed only when the file to download is a binary data format (*.bin). Specify the start address from which a specified file is downloaded.	
	Default	0
	Method to change	By entering directly from the keyboard
	Specifiable values	Hexadecimal number in the range 0x0 to 0xFFFFFFFF

Remark Specification of whether or not to download object information or symbol information is accepted only when the type of file to download is a load module file.

(c) Confirming the order in which a download is executed

A download is executed in the order in which files are displayed in the [\[Download File List\] area](#).
To change the order, use the [Up] or [Down] button to move any file up or down in the list.

(d) Clicking the [OK] button

Settings you've made in this dialog box are enabled, with the specified file added as a download file (The added file name and its download condition are displayed in the [\[Download\]](#) category on the [Property panel's \[Download File Settings\] tab](#)).

(3) Downloading multiple load module files

To download multiple load module files, follow the procedure described below as you fill in the [Download Files dialog box](#).

Caution When debugging a program comprised of multiple load module files, be careful that location addresses will not overlap.

(a) Clicking the [Add] button

When you click the [Add] button, a blank item ("-") is displayed on the last line in the [\[Download File List\] area](#).

(b) Setting the properties of a download file to add

In the [\[Download file property\] area](#), select a download file you want to add and set its download conditions. For each displayed item, make the following setting.

When setting is completed, the file name you've specified here is reflected in the blank item in the [\[Download File List\] area](#).

File	Specify a file in load module format to add (specifiable in up to 259 characters).	
	Default	Blank
	Method to change	By entering directly from the keyboard, or specifying in the Select Download File dialog box that is opened by clicking the [...] button displayed at the right edge of this property when it is selected
	Specifiable values	See "Table 2-2. Downloadable File Formats"
File type	Specify the file type of a download file to add. Here, select [Load module file].	
	Default	Load module file
Download object	Specify whether or not to download object information from a specified file.	
	Default	Yes
	Method to change	By selecting from drop-down list
	Specifiable values	Yes Downloads object information.
		No Does not download object information.
Download symbol information	Specify whether or not to download symbol information from a specified file ^{Note 1} .	
	Default	Yes
	Method to change	By selecting from drop-down list
	Specifiable values	Yes Downloads symbol information.
		No Does not download symbol information.
Specify the PIC/PID offset	Specify whether or not to change the positions of PIC (Position Independent Code) and PID (Position Independent Data) areas of the load module to be downloaded from those at which they were when the load module was created. When this item is changed to [Yes], "PIC Offset" and "PID Offset" are displayed in the sub-item.	
	Default	No
	Method to change	By selecting from drop-down list
	Specifiable values	Yes Specifies PIC/PID offset ^{Note 2} .
		No Does not specify PIC/PID offset.

PIC Offset	Enter an offset value from an address that was in the load module when it was created. For example, if the program section's start address begins with the address 0x1000, enter "1000" for this item. The subject section is downloaded to the address 0x2000.	
	Default	0
	Method to change	By entering directly from the keyboard
	Specifiable values	Hexadecimal number in the range 0x0 to 0xFFFFFFFF
PID Offset	Enter an offset value to be set in the PID register that was specified in the load module when it was created. For example, to set 0x200 in the PID register when the load module is executed, enter "200" for this item.	
	Default	0
	Method to change	By entering directly from the keyboard
	Specifiable values	Hexadecimal number in the range 0x0 to 0xFFFFFFFF

- Notes**
1. Unless symbol information is downloaded, source-level debugging cannot be performed.
 2. If you select [Yes] for load modules created not using the PIC/PID functions (see Section "2.7 Usage of PIC/PID Function"), debug operation cannot be guaranteed.

Remark For the load module files that do not require symbol information, you can set [No] for the [[Download symbol information](#)] item to reduce the amount of memory used (However, this file cannot be debugged at the source level.).

(c) Confirming the order in which a download is executed

A download is executed in the order in which files are displayed in the [[Download File List](#)] area.
To change the order, use the [Up] or [Down] button to move any file up or down in the list.

(d) Clicking the [OK] button

Settings you've made in this dialog box are enabled, with the specified load module file added as a download file. (The added file is displayed in the [[Download](#)] category on the [Property panel's \[Download File Settings\] tab](#)).

(4) Performing source-level debug with hex format, Motorola S format, or binary data format files at the source level

Even when a file in hex format (*.hex), Motorola S format (*.mot), or binary data format (*.bin) is specified to be the subject file to download, it is possible to do source-level debug by downloading symbol information for the load module file from which the subject file was created, along with the subject file that you download.
In this case, follow the procedure described below as you fill in the [Download Files dialog box](#).

(a) Clicking the [Add] button

When you click the [Add] button, a blank item ("-") is displayed on the last line in the [[Download File List](#)] area.

(b) Setting the properties of a load module file

In the [[Download file property](#)] area, specify for each item as shown below.

File	Specify a load module file from which the file in hex format (*.hex), Motorola S format (*.mot), or binary data format (*.bin) that you want to download was created. Enter directly from the keyboard, or specify in the Select Download File dialog box that is opened by clicking the [...] button.
File type	Specify [Load module file] (default).
Download object	Specify [No].
Download symbol information	Specify [Yes] (default).
Specify the PIC/PID offset	Specify [No] (default) ^{Note} .

Note For files in hex format, Motorola S format, or binary data format, be aware that the destination to which they are downloaded cannot be discriminated between the PIC and PID areas. If you specify PIC/PID offsets in a load module file, the result will be that the specified offsets do not agree with the debug function and the file cannot be debugged.

(c) Clicking the [OK] button

Settings you've made in this dialog box are enabled, with the specified load module file added as a download file (Only the symbol information included in the load module file becomes the subject to download.).

(5) Downloading files to external flash memory [E1] [E20]

The download files you've specified in the [Download Files dialog box](#) can be downloaded to the flash memory connected to an external bus of the microcontroller used (i.e., the external flash memory).

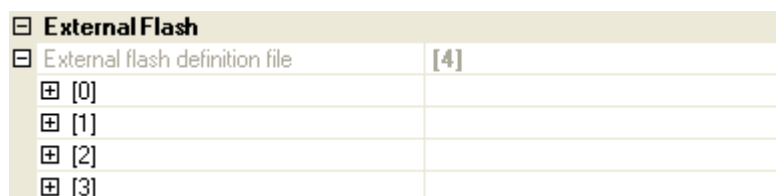
In this case, follow the procedure described below.

(a) Setting up the properties panel

In the [\[External Flash\] \[E1\] \[E20\]](#) category on the [Property panel's \[Debug Tool Settings\] tab](#), register an external flash definition file (USD file).

External flash memory is recognized by registering a USD file in the [\[External Flash\] \[E1\] \[E20\]](#) category. Up to four USD files can be registered.

Figure 2-61. [External Flash] Category



<1> [External Flash Definition File]

Shows the maximum number of registrable USD files.

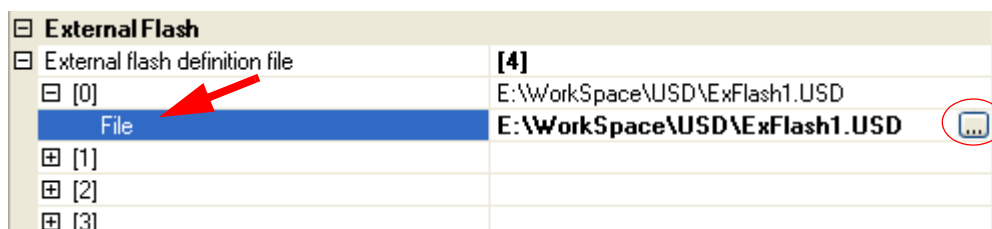
To register a USD file, specify a USD file name in the [File] property that is expanded on display as a sub-property.

Enter a file name directly (if specified by a relative path, it should be referenced to the project folder), or select a file in the [External flash memory dialog box \[E1\] \[E20\]](#) that is opened by clicking the [...] button, which is displayed at the right edge in the setup column of this property when it is selected.

To delete registration of a USD file, move the caret to the file name on the relevant [File] property and then click the [Delete] key.

For more information on USD files, visit Renesas website and see the user's manual for the External Flash Memory Editor.

Figure 2-62. Opening the External flash memory Dialog Box

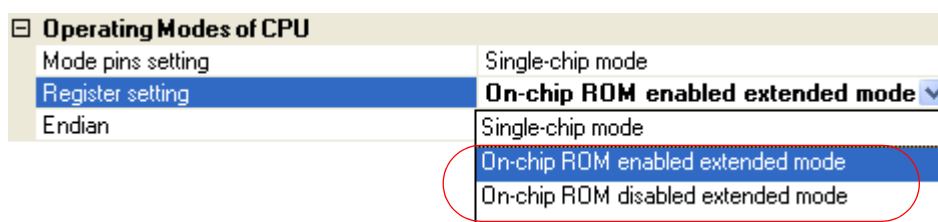


Caution The endian information in USD files is not reflected on the [Memory mappings] property in the [Memory] category on the Property panel's [Debug Tool Settings] tab. Therefore, the endian in the external area to which they are downloaded should be changed to suit the endian information in USD files.

(b) Setting the CPU operation mode

For files to be downloaded to external flash memory, it is necessary to set the microcontroller's operation mode when connecting with the debug tool. To do this, change the [Register setting] property in the [Operating Modes of CPU] [E1] [E20] category on the Property panel's [Debug Tool Settings] tab by selecting [On-chip ROM enabled extended mode] or [On-chip ROM disabled extended mode] from the pulldown list, as shown below.

Figure 2-63. Specifying the CPU Operation Mode



(c) Executing a download

Execute a download (see "(3) Executing a download").

- Cautions**
1. External flash memory cannot be rewritten on the Memory panel.
 2. To use a flash memory device that does not support the use of the Lock command, specify that the lock bit be cleared when you generate USD files. This specification helps you omit an unnecessary process to confirm the lock bit status.

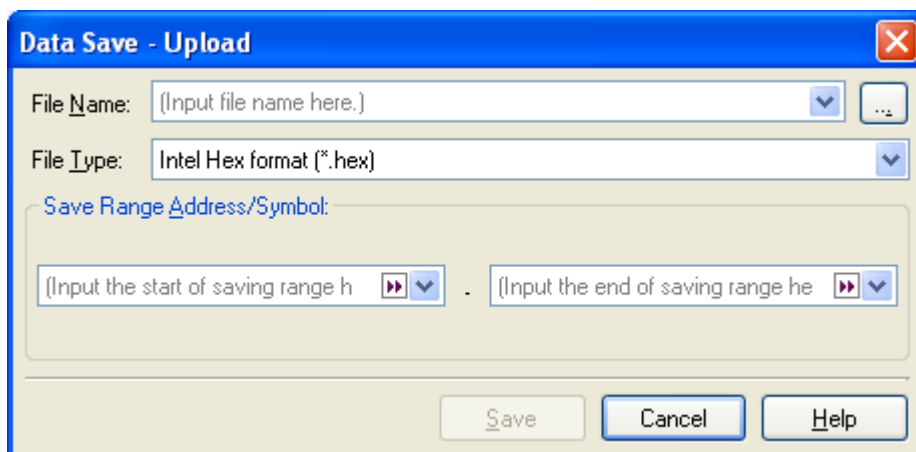
2.5.3 Executing an upload

The memory content of the currently connected debug tool can be saved (uploaded) to any file.

To do an upload, use the Data Save dialog box that is opened by choosing [Upload...] from the [Debug] menu.

Follow the procedure described below to make the necessary setting in this dialog box.

Figure 2-64. Uploading Memory Contents (Data Save Dialog Box)

**(1) Specifying the [File Name]**

Specify a file name in which you save.

Enter directly in the text box (specifiable in up to 259 characters), or select an input history item from the drop-down list (up to 10 history entries).

Also, you can select a file from the [Select Data Save File dialog box](#) that is opened by clicking the [...] button.

(2) Specifying the [File Type]

Select the type of file in which you want to save from the drop-down list below.

The selectable file types are as follows:

Note that the [ID Tagged] item is displayed only when the selected microcontroller is a product that comes with internal data flash memory and supports the ID tag function. By selecting the [ID Tagged] item, it is possible to save information on ID tag.

Table 2-3. Uploadable File Formats

List display	Format
Intel Hex format (*.hex)	Hex format ^{Note}
Motorola S-format (*.mot)	S Record format
Binary data (*.bin)	Binary data format

Note In the Intel hex format, memory contents are always saved in "extension linear address code (32-bit address)."

(3) Specifying the [Save Range Address/Symbol]

Specify the "start address" and "end address" to set a range of memory to be saved in a file.

Enter hexadecimal values or address expressions directly in the respective text boxes or select an input history item from the drop-down list (up to 10 history entries).

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see ["2.20.2 Symbol name completion function"](#)).

(4) Clicking the [Save] button

Memory contents are saved in specified form to a specified file as upload data.

2.6 Displaying and Changing Programs

This section describes how to display and change programs when a load module file with the debug information is downloaded to a debug tool.

Downloaded programs can be displayed in the following panels.

- [Editor panel](#)

The source file is displayed and can be edited.

Furthermore, the source level debugging (see "[2.9.3 Execute programs in steps](#)") and the display of the code coverage measurement result (see "[2.15.2 Display coverage measurement results](#)") can be performed in this panel.

- [Disassemble panel](#)

The result of disassembling the downloaded program (the memory contents) is displayed and can be edited (line assemble).

Furthermore, the instruction level debugging (see "[2.9.3 Execute programs in steps](#)") and the display of the code coverage measurement result (see "[2.15.2 Display coverage measurement results](#)") can be performed in this panel. In this panel, the disassemble results can be displayed with the corresponding source text (default).

Remark It is normally necessary to download a load module file with debugging information in order to perform the source level debugging, but it is also possible to do so by downloading a hex format (*.hex), Motorola S format (*.mot), or binary data format (*.bin) file (see "[\(4\) Performing source-level debug with hex format, Motorola S format, or binary data format files at the source level](#)").

2.6.1 Displaying source files

Use the [Editor panel](#) shown below to display the source file.

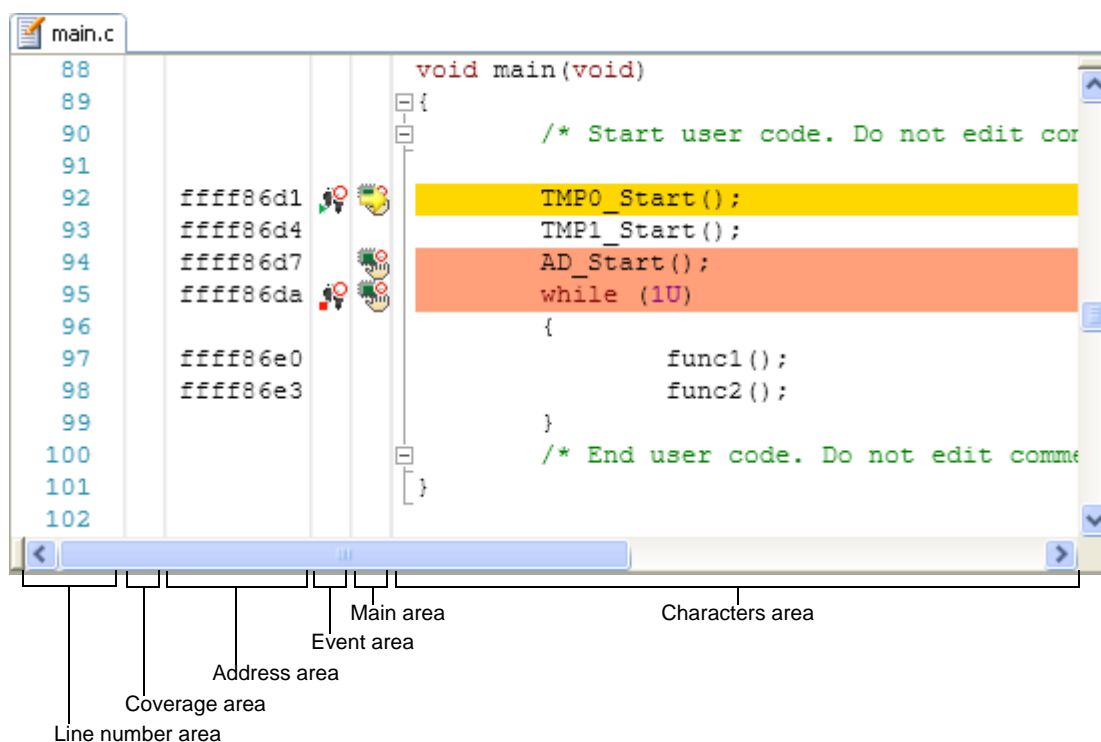
The Editor panel automatically opens with displaying source text of the specified position (see "[2.5.1 Execute downloading](#)") when a load module file is successfully downloaded.

If you want to open the Editor panel manually, double-click on the source file in the [Project Tree panel](#).

For details on the contents and function in each area, see the section for the [Editor panel](#).

- Remarks 1.** You can open a file with a specific encoding selected in the [Encoding dialog box](#) that is opened via the [File] menu >> [Open with encoding...].
- 2.** You can zoom in and out of this panel (See "[\(I\) Zoom in or out on a view](#)") by using the [Ctrl] key + mouse-wheel combination.

Figure 2-65. Displaying the Source File (Editor Panel)



This section describes the following.

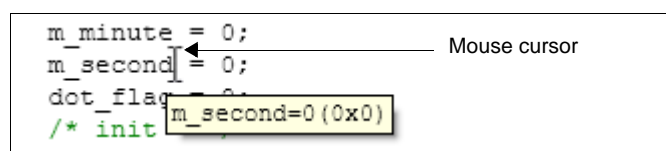
- (1) [Displaying variables](#)
- (2) [Searching for characters](#)
- (3) [Moving to a specified line](#)
- (4) [Jump to functions](#)
- (5) [Jump to a desired line \(tag jump\)](#)

(1) Displaying variables


When hovering the mouse cursor over a variable in the source text, a pop-up that shows the name and value of the variable is displayed ("*<variable name>=<variable value>*").

The display format of the variable value is same as "Table A-10. Display Format of Watch Expressions (Default)" depending on the type of the variable.

Figure 2-66. Pop-up Display of Variables (Editor Panel)

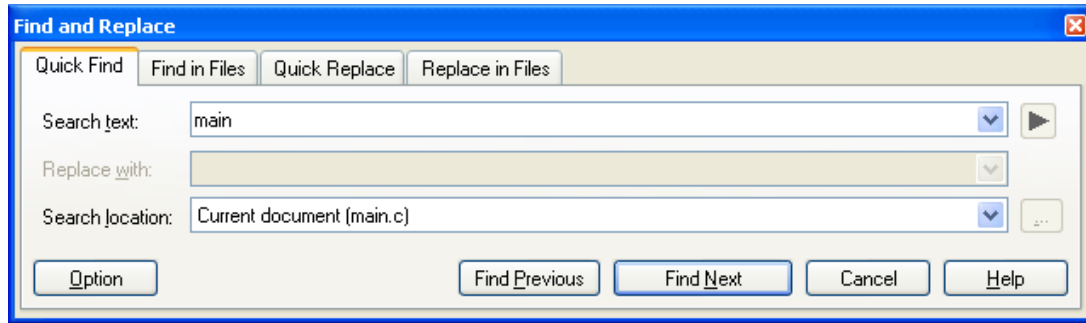


(2) Searching for characters

To search for memory values, use the Find and Replace dialog box that is opened by selecting the  button on the toolbar.

In this dialog box, follow the procedure described below to search for characters.

Figure 2-67. Character Search in Source Text (Find and Replace Dialog Box)

**(a) Specifying [Search text]**

Enter characters to search.

A word (variable/function) at the caret position in the [Editor panel](#) is specified by default.

If you want to change it, directly enter the characters into the text box (up to 1024 characters) or select an input history item from the drop-down list (up to 10 items).

(b) Specifying [Search location]

Select [Current document (*file name*)] from the drop-down list.

(c) Clicking the [Find Previous]/[Find Next] button

When the [Find Previous] button is clicked, search will start in the order from the large address number to small and the search results are displayed selected in the [Editor panel](#).

When the [Find Next] button is clicked, search will start in the order from the small address number to large and the search results are displayed selected in the Editor panel.

Remarks 1. Click the [Option] button to specify to use wild card, case sensitivity, word by word search, and so on.

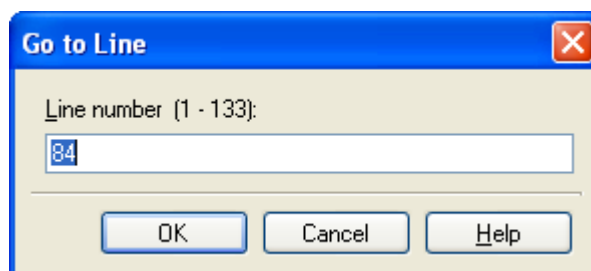
2. In the Find and Replace dialog box, various search/replace operation can be performed by selecting [Find in Files], [Quick Replace], or [Replace in Files] tab.

(3) Moving to a specified line

To move to a specified address in the source text, select [Go To...] on context menu and use the [Go to Line dialog box](#) that is opened.

In this dialog box, follow the procedure described below to move to a specified line.

Figure 2-68. Move to the Specified Line in Source Text (Go to Line Dialog Box)



(a) Specifying [Line number (valid line range)]

"(valid line range)" shows the range of valid lines in the current file.

Directly enter a decimal value as the number of the line you want to move the caret to.

You can also enter a symbol in this area.

By default, the number of the line where the caret is currently located in the Editor panel is displayed.

(b) Clicking the [OK] button

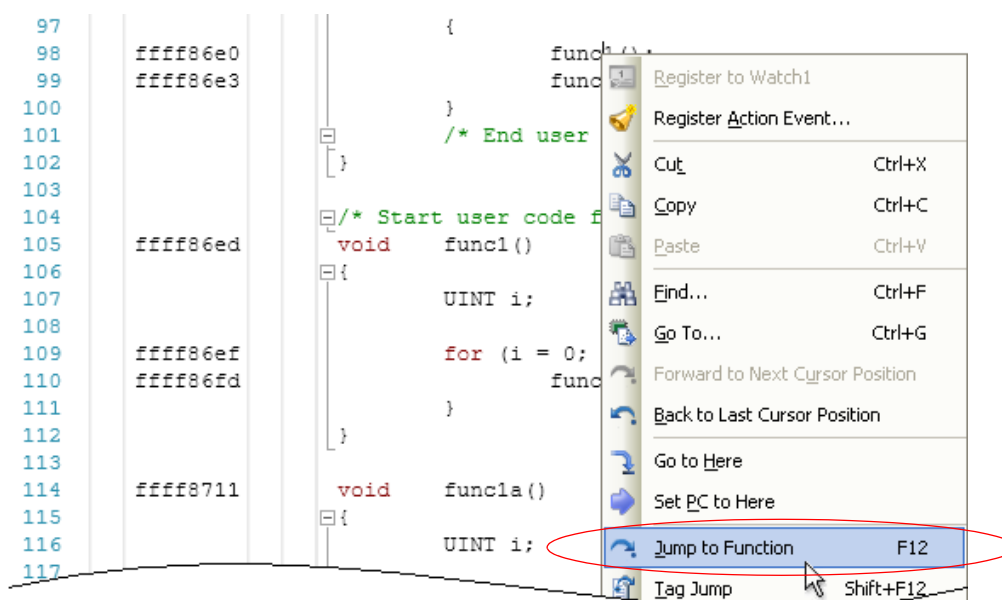
The caret moves to the specified line.

(4) Jump to functions

It automatically recognizes the currently selected characters or the word at the caret position as the function name and jumps to the first executable line of the target function.

Select [Jump to Function] from the context menu after moving the caret to the target function on the source text.

Figure 2-69. Jump to Functions



Note that this function is available only when the following conditions are satisfied for each specific build tool.

(a) When CC-RX is used

- The target is a function, variable, or label in C language.
- The focus is in the [Editor panel](#).

(b) When an external build tool is used

- The target function^{Note 1} resides in an active project.
- The focus is in the [Editor panel](#).
- A file^{Note 2} with the symbol information is selected for [\[Download files\]](#). In case it is disconnected from the debug tool, the above file is specified as the first file in [\[Download files\]](#).

Notes 1. A jump to a static function cannot be made when the debug tool is disconnected.

- 2.** When the file is in the hex format, setting for downloading the symbol information is required (see "(4) [Performing source-level debug with hex format, Motorola S format, or binary data format files at the source level](#)").

Caution When multiple statements are described in a line, a jump to an illegal location may be made.

Remark The judgement of words will depend on the build tool being used.

(5) Jump to a desired line (tag jump)

If the information of a file name, a line number and a column number exist in the line at the caret position, you can open the file in another [Editor panel](#) and jump to the corresponding line and the corresponding column (if the Editor panel is already open, you can jump to the panel).

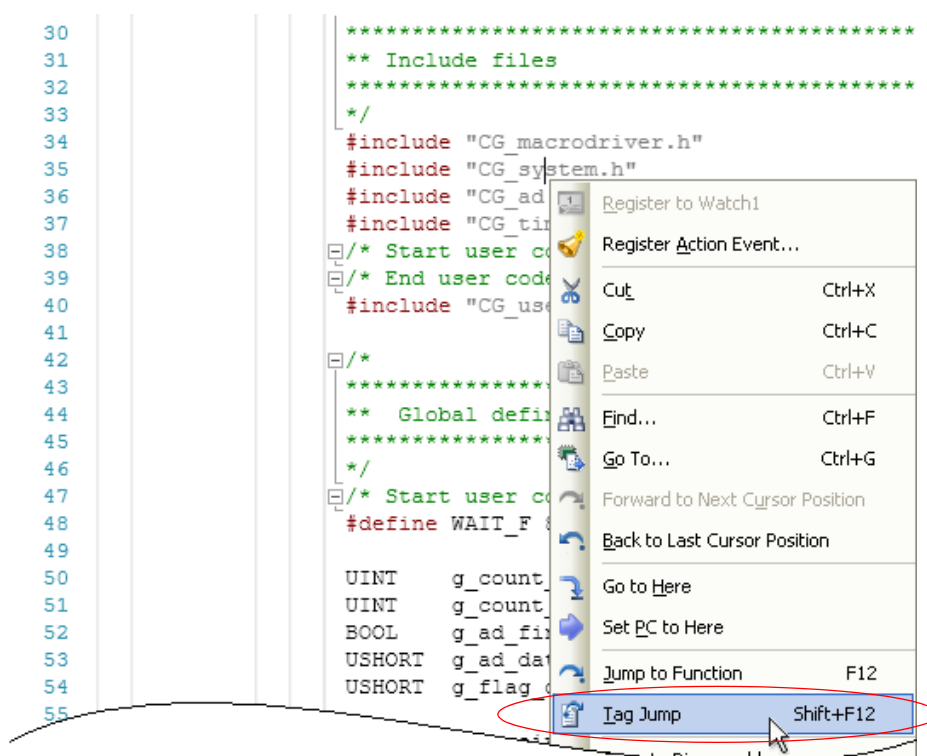
Select [Tag Jump] from the context menu after moving the caret to the line on the source text.

The tag jump is operated as follows:

Table 2-4. Operation of Tag Jump

Example of Character String	Operation
C:\work\src.c	Jumps to the top line of the file "C:\work\src.c".
Tmp\src.c	Jumps to the top line of the file ".Tmp\src.c".
C:\work\src.c(10)	Jumps to the tenth line from the top of the file "C:\work\src.c".
C:\work sub\src.c"(10)	Jumps to the tenth line from the top of the file "C:\work sub\src.c".
C:\work\src.c(10,5)	Jumps to the fifth column of the tenth line from the top of the file "C:\work\src.c".

Figure 2-70. Tag Jump



- Remarks**
1. Jumps are case-insensitive.
 2. The reference point of the path is the project folder in which the file is registered. If the file is not registered in any project, the reference point of the path will be the active folder.
 3. Path specifications (path/file names) including space characters must be enclosed in "".

2.6.2 Displaying the disassembled result

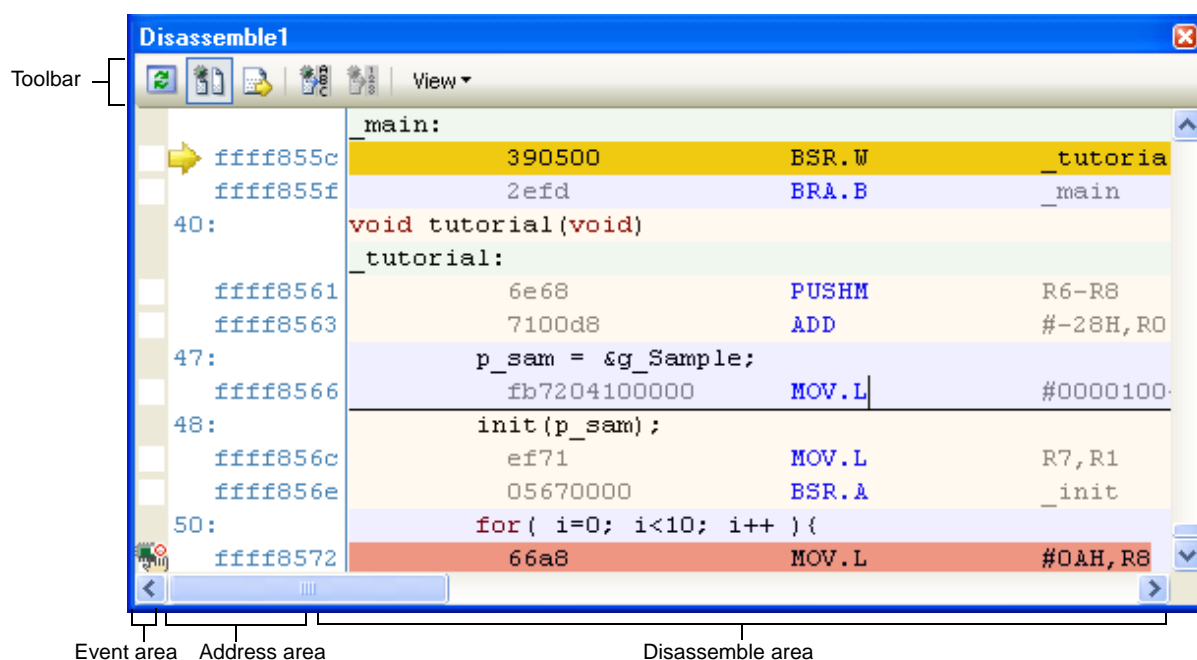
To display the disassembled result (disassembled text) of a downloaded program (memory content), use the [Disassemble panel](#) shown below.


Choose [Disassemble] from the [View] menu and then select [Disassemble1 - 4].

Up to four disassemble panels can be opened at a time, with each panel discriminated by the name in their title bar, "Disassemble1," "Disassemble2," "Disassemble3," or "Disassemble4."

For details on how to read each area and details about their functionality, see the section in which the [Disassemble panel](#) is described.

Figure 2-71. Displaying the Disassembled Result (Disassemble Panel)



Remark In the [Scroll Range Settings dialog box](#) which opens by clicking on [View] >> , in the toolbar, you can set the scroll range of the vertical scroll bar on this panel.

Following methods of operation are described here.

- (1) [Changing the display mode](#)
- (2) [Changing the display form](#)
- (3) [Moving to a specified address](#)
- (4) [Moving to a symbol definition part](#)
- (5) [Saving the displayed contents of disassembled results](#)

(1) Changing the display mode

Disassembled results are displayed, by default, in the mixed display mode (a mixed display of disassembled text and source text).


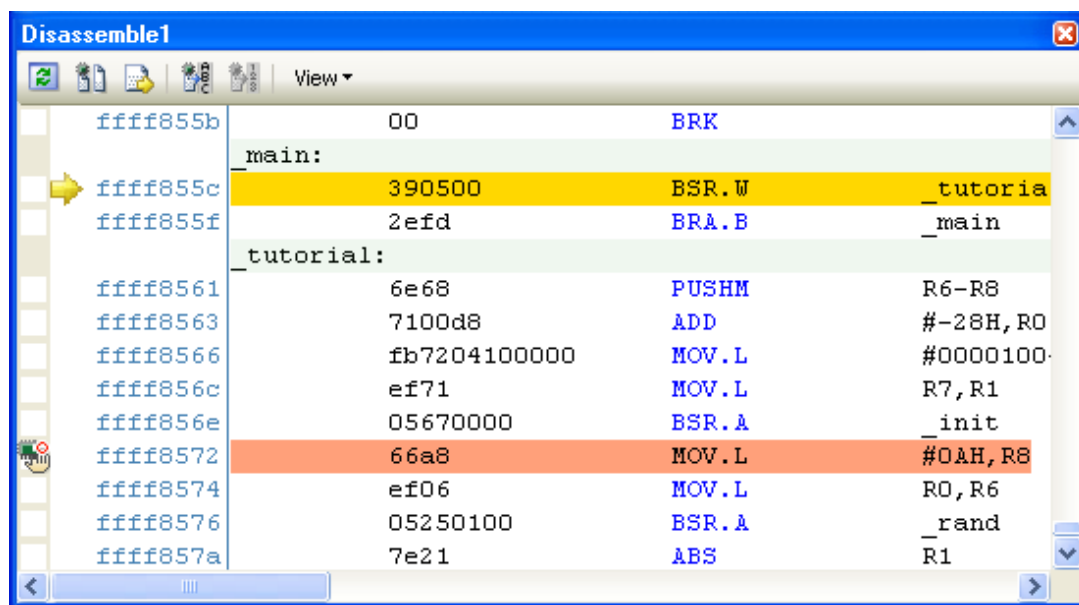


By clicking the toolbar button  (toggle), you can choose to show or not show the source text.

Figure 2-72. Example Display Where Source Text is Hidden

**(2) Changing the display form**

The form in which disassembled results are displayed can be changed freely by using the toolbar buttons shown below.

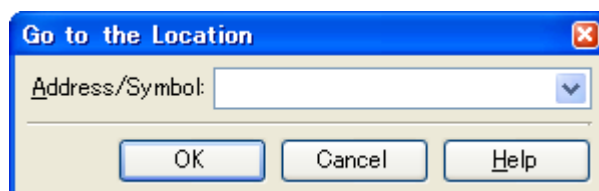
View	Shows the following buttons that change the display form.
	Displays the offset values of labels. If the address has no labels defined, an offset value from the closest label is displayed.
	Displays address values in the form "symbol + offset value" (default). However, if the address value has a symbol defined, only the symbol is displayed.

(3) Moving to a specified address

To move to a specified address in the disassembled text, select [Go to...] on context menu and use the [Go to the Location dialog box](#) that is opened.

In this dialog box, follow the procedure described below to Move to a specified address.

Figure 2-73. Move to the Specified Address in Disassembled Text (Go to the Location Dialog Box)

**(a) Specifying [Address/Symbol]**

Specify an address to which you want to move the caret.

Enter an address directly in the text box or select an input history item from the drop-down list (up to 10 history entries).


Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").


(b) Clicking the [OK] button

The caret moves to the specified address.

(4) Moving to a symbol definition part

The caret position can be moved to the address at which a symbol is defined.

Move the caret to an instruction that is referencing the symbol and then click the toolbar button .

Also, if, subsequently to the above operation, you click the toolbar  button, the caret position is returned to the instruction that was referencing the symbol before the caret was moved.

(5) Saving the displayed contents of disassembled results

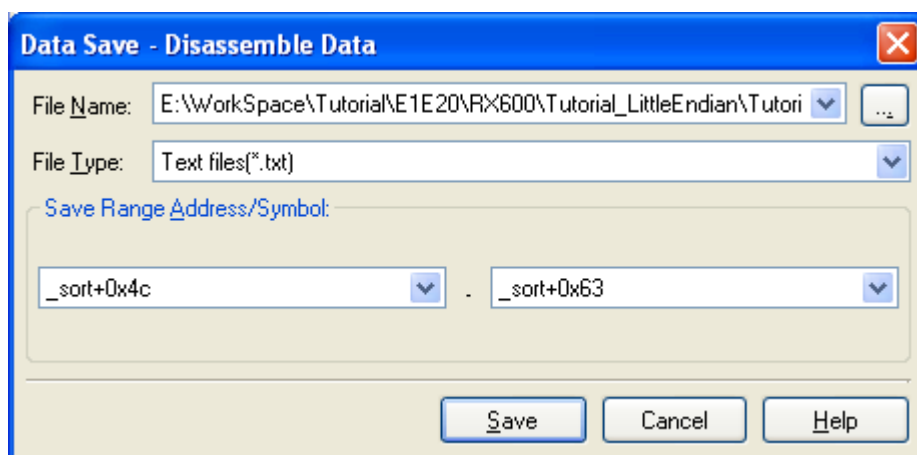
The contents of disassembled results can be saved to a text file (*.txt) or CSV file (*.csv).

When saving to a file, CubeSuite+ gets latest information from the debug tool and saves it in the form in which data are displayed on this panel.

Choose [Save Disassembled Data As ...] from the [File] menu, and the [Data Save dialog box](#) shown below is opened. (At this time, if a range is selected on panel while you perform this operation, it is possible to save only the selected range of disassembled data.)

In this dialog box, follow the procedure described below to save the displayed contents of disassembled results.

Figure 2-74. Saving Disassembled Result (Data Save Dialog Box)

**(a) Specifying the [File Name]**

Specify a file name in which you want to save.

Enter it directly in the text box (specifiable in up to 259 characters) or select an input history item from the drop-down list (up to 10 history entries).

Also, you can select a file using the [Select Data Save File dialog box](#) that is opened by clicking the [...] button.

(b) Specifying the [File Type]

Select the type of file in which you want to save from the drop-down list below.

The selectable file types are as follows.

List display	Format
Text files (*.txt)	Text format (default)
CSV (Comma-Separated) (*.csv)	CSV format ^{Note}

Note Each piece of data are separated with a comma (,) when saved.

To avoid the problem of an invalid file format in cases where any data includes a comma (,), each piece of data are enclosed in double-quotes (" ") when they are output to a file.

(c) Specifying the [Save Range Address/ Symbol]

Specify the "start address" and "end address" to set a range of data to be saved in a file.

Directly enter hexadecimal values or address expressions in the respective text boxes or select an input history item from the drop-down list (up to 10 history entries).

Note that if a range is selected on panel, this selected range is specified, by default, in the text boxes. If no range is selected, the currently displayed range on panel is specified.

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 [Symbol name completion function](#)").

(d) Clicking the [Save] button

Disassembled data is saved in specified format to a specified file.

Figure 2-75. Disassembled Data Output Image when Saved

Label (symbol name)	:			← Label (symbol) line
File name	:	Line number	:	C language source text
:	:	:	:	← Source text line
Address	:	Offset	:	Code
:	:	:	:	Result of Disassembling
:	:	:	:	← Disassembling line

Remarks 1. If panel contents are saved over an existing file by selecting [Save Disassembled Data] on [File] menu, the respective [Disassemble panels](#) (disassembly 1-4) are handled individually.

Also, as to the save range, the previously specified address range is applied when data is saved.

2. By selecting [File] >> [Print...], you can print the image currently displayed on this panel.

2.6.3 Executing a build in parallel with other processes

CubeSuite+ provides a facility to automatically start a build in synchronism with the timing described below (rapid build function).

(1) For other than the debug-only project

- When any of the C source files, C++ source files, assembler source files, header files, link directive files, symbol information files, object module files, relocatable module files or library files added to the project is updated
- When a file to build is added or removed from the project
- When the order in which object module files is linked is changed
- When the properties of the build tool or the file to build are changed

(2) For the debug-only project

- When a C source file, C++ source file, assembler source file, or header file added to the debug-only project is saved after editing
- When a C source file, C++ source file, assembler source file, or header file is added or removed from the debug-only project
- When properties of the debug-only project are changed

By enabling the rapid build function, it is possible to perform a build in parallel with the above operation.

To choose to enable or disable the rapid build function, click [Rapid Build] on the [Build] menu (This is a toggle switch, which is, by default, enabled.).

Caution For this function to be enabled when you use an external editor, it is necessary to check the [\[Observe registered files changing\]](#) checkbox in the [\[General - Build/Debug\]](#) category of the [Option dialog box](#).

- Remarks 1.** It is recommended that after editing the source file, you should always be sure to press the [Ctrl] + [S] keys to save it each time.
2. When you've chosen to enable or disable the rapid build function, your selection is applied to the entire project (main project and sub-projects).
 3. If, while a rapid build is under execution, the rapid build function is switched off (disabled), execution of the rapid build is aborted on the spot.

2.6.4 Performing line assembly

The instructions and instruction codes displayed on the [Disassemble panel](#) can be edited (line-assembled). Following methods of operation are described here.

- (1) [Editing instructions](#)
- (2) [Editing instruction code](#)

(1) Editing instructions

Edit the character string of the instruction directly from the keyboard.

(a) Switching to the edit mode

Double-click an instruction you want to edit, or while the caret is moved to the subject instruction, select [\[Edit Disassemble\]](#) on context menu, and the subject you're going to edit is placed into the edit mode.

(b) Editing instructions

Use keyboard to directly edit the instructions.

(c) Writing into memory

Press the [Enter] key when you've finished editing, and the altered instruction is automatically line-assembled and the resulting code is written into memory.

However, if this alteration results in an invalid instruction, the character string you've edited is displayed in red color and not written into memory.

Note that if space is created in memory by overwriting the disassembled result being displayed on panel with another instruction, bytes are automatically compensate for with nop instructions as in the example shown below.

Examples 1. When the ADD instruction on third line (6-byte instruction) is overwritten with a BCLR instruction (4-byte instruction)

Before editing	5020	AND	[R2].UB,R0
	99e0	MOV.W	0CH[R6],R0
	700fb98a322a	ADD	#2A328AB9H,R0,R15
	5327	AND	R2,R7

After editing	5020	AND	[R2].UB,R0
	99e0	MOV.W	0CH[R6],R0
	f22a45b2	BCLR	#2,0B245H[R2]
	03	NOP	
	03	NOP	
	5327	AND	R2,R7

2. When an MOV.W instruction on second line (2-byte instruction) is overwritten with a BCLR instruction (4-byte instruction)

Before editing	5020	AND	[R2].UB,R0
	99e0	MOV.W	0CH[R6],R0
	700fb98a322a	ADD	#2A328AB9H,R0,R15
	5327	AND	R2,R7
After editing	5020	AND	[R2].UB,R0
	99e0	MOV.W	0CH[R6],R0
	f22a45b2	BCLR	#2,0B245H[R2]
	03	NOP	
	03	NOP	
	03	NOP	
	03	NOP	
	5327	AND	R2,R7

(2) Editing instruction code

To edit instruction code, follow the procedure described below.

(a) Switching to the edit mode

Double-click instruction code you want to edit, or while the caret is moved to the subject instruction code, select [\[Edit Code\]](#) on context menu, and the subject you're going to edit is placed into the edit mode.

(b) Editing instruction code

Edit the character string of instruction code directly from the keyboard.

(c) Writing into memory

Press the [Enter] key when you've finished editing, and instruction code is written into memory.

However, if this alteration results in an invalid instruction, the character string you've edited is displayed in red color and not written into memory.

When instruction code is written into memory, the disassembled result is updated at the same time.

2.7 Usage of PIC/PID Function

The PIC/PID function enables the code and data in the ROM to be reallocated to desired addresses without re-linkage even when the allocation addresses have been determined through previously completed linkage.

To use the PIC/PID function, a "master" program and an "application" program must be prepared.

In the PIC/PID function, a program whose code or data in the ROM has been converted into PIC or PID is called an application, and the program necessary to execute an application is called the master.

This section describes debugging of an application program (load module) whose code or data in the ROM has been converted into PIC or PID and reallocated to different addresses.

- PIC

When the pic option is specified for compilation, the PIC function is enabled and the code in the code area (P section) becomes PIC. The PIC always uses PC relative mode to acquire branch destination addresses or function addresses, so it can be reallocated to any desired addresses even after linkage.

- PID

When the pid option is specified for compilation, the PID function is enabled and the data in ROM data areas (C, C_2, C_1, W, W_2, W_1, and L sections) becomes PID. A program executes relative access to the PID by using the register (PID register) that indicates the start address of the PID. The user can move the PID to any desired addresses by modifying the PID register value even after linkage.

- Remarks 1.** For details on the PIC/PID function, see chapter 7, Startup, in "CubeSuite+ for the RX: Coding".
- 2.** For setting of the PIC/PID function by the build tool, see Chapter 2, Functions, in "CubeSuite+ for the RX: Build".

To start debugging after changing the allocation of a load module whose code or data in the ROM has been converted into PIC or PID, take the following steps.

(1) Add a download file

Add the application load-module file as a download file for the master (see ["2.7.1 Changing the allocation of a load module using the PIC/PID function"](#)).

(2) Set conditions for downloading of the load module file

Specify two offset values for the application load module: [PIC offset], which is an offset from the original address, and [PID offset], which is an offset to be set in the PID register selected at the time the load module was created.

(3) Download

Download the master and the application load-module file (see ["2.5.1 Execute downloading"](#)).

Debugging of the code and data in the ROM allocated to new addresses is now possible.

2.7.1 Changing the allocation of a load module using the PIC/PID function

An application load-module file can be added to the master through the [Download files] property under the [Download] category on the [Download File Settings] tab in the Property panel.

Figure 2-76. [Download] Category



- [Download files]

Click on the [...] button to open the Download Files dialog box.

Click on the [Add] button in the [Download File List] area of the Download Files dialog box and set up information on the application load module in the [Download file property] area.

- [File]

Specify the application load-module file to be downloaded.

- [Specify the PIC/PID offset]

Select [Yes]. [PIC offset] and [PID offset] will appear.

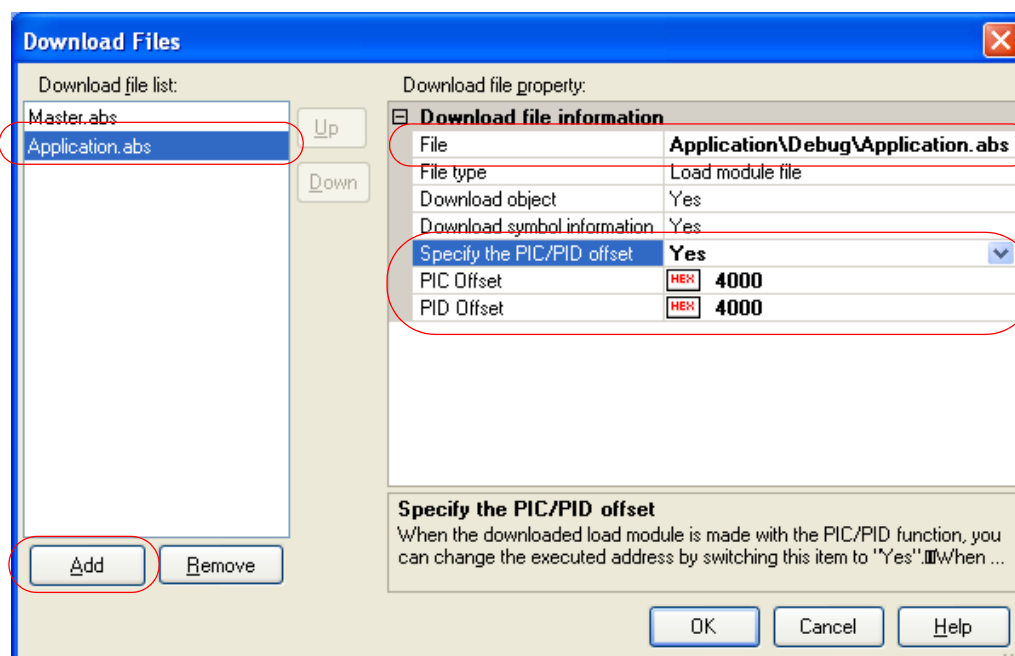
- [PIC offset]

Specify an offset from the original address allocated at the time the load module was created.

- [PID offset]

Specify an offset to be set in the PID register selected at the time the load module was created.

Figure 2-77. Adding a Download File and Changing Conditions for Downloading (Download Files Dialog Box)

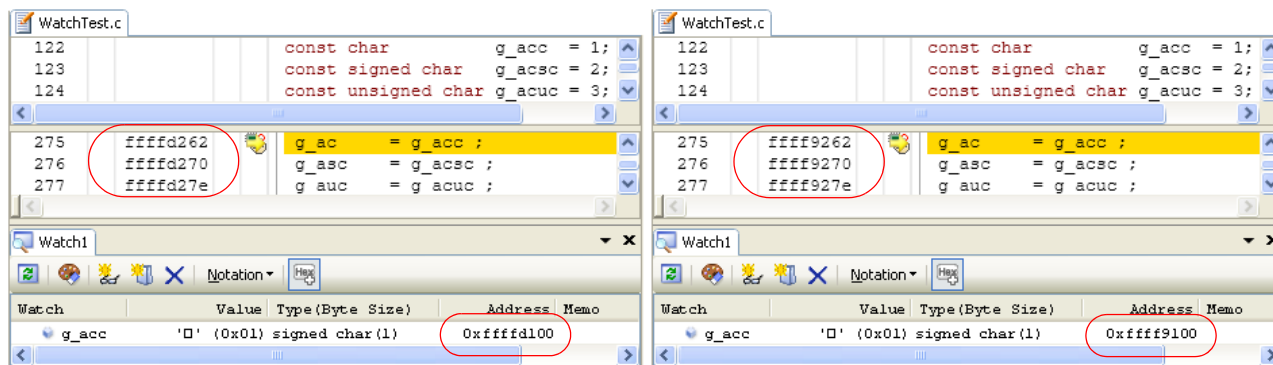


When the load module is downloaded after the values of [PIC offset] and [PID offset] have been changed, the allocation of addresses for the P section and external or static variables is changed as shown in the figure below.

The left half shows an example where 4000 is specified for [PIC offset] and [PID offset] in the [Download Files dialog box](#) while the right half shows an example where 0 is specified.

In the left half, 4000 is added to the original address.

Figure 2-78. Example of Downloading after the Offset Values of [PIC offset] and [PID offset] Have been Changed



2.8 Setting Overlay Sections

The optimizing linkage editor (OptLnk) used by the CC-RX provides the start option, which allows two or more sections defined in a program to be allocated to a single address. Sections that have been allocated in this way are called overlay sections.

This section describes how to set up the "overlay-section selection facility" for overlay sections in a load module.

When the program is executed, only one of the sections allocated to the same address in the load module is executed.

The debug tool provides the "overlay-section selection facility" to select a particular overlay section to be debugged preferentially (hereafter referred to as the priority section).

With the priority section selected before execution of the program, debugging of that section is simple because debugging information of other sections is hidden.

An example of using OptLnk's start option to define overlay sections is given below.

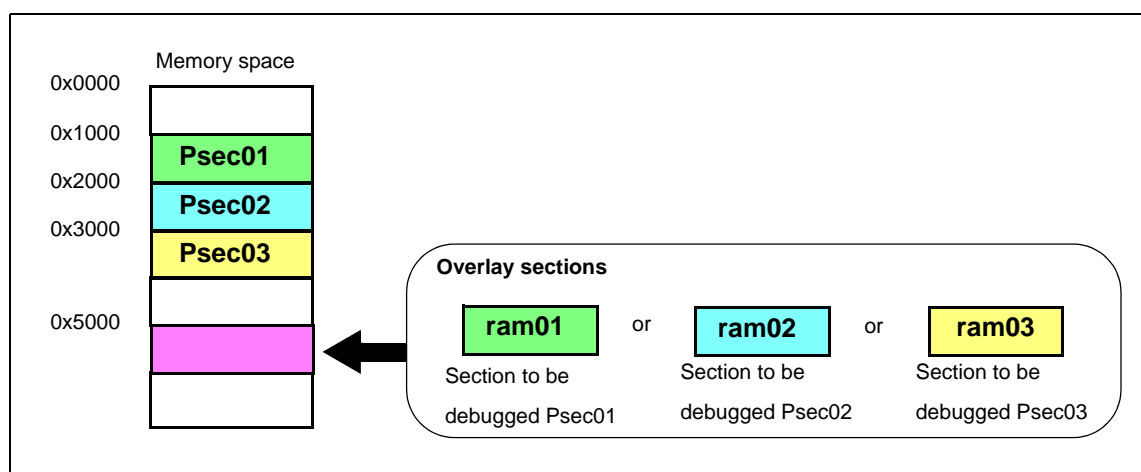
Example Defining overlay sections by OptLnk

```
start = Psect01/1000,Psect02/2000,Psect03/3000,ram01:ram02:ram03/5000
rom = Psect01=ram01,Psect02=ram02,Psect03=ram03
```

The rom option reserves section "ramXX", which is as large as the "PsectXX" section, and relocates the symbols defined in the "PsectXX" section to the corresponding addresses in the "ramXX" section (XX: 01, 02, or 03 in this example).

Remark For setting of overlay sections by the build tool, refer to Chapter 2, Functions, in "CubeSuite+ for the RX: Build".

Figure 2-79. Allocation of Overlay Sections



2.8.1 Selecting the priority section

The priority section can be selected under the [\[Debug Information\]](#) category on the [\[Download File Settings\]](#) tab in the [Property panel](#).

Figure 2-80. [\[Debug Information\]](#) Category

Debug Information	
Execute to the specified symbol after CPU Reset	Yes
Specified symbol	_main
Specify the debugged overlay section	Yes
Overlay sections	[3]

(1) [\[Specify the debugged overlay section\]](#)

"Yes" is displayed when overlay sections exist in the load module (this cannot be changed).

(2) [\[Overlay sections\]](#)

Address groups where overlay sections exist are displayed.

This property only appears when "Yes" is displayed on the [\[Specify the debugged overlay section\]](#) property.

Figure 2-81. [\[Overlay sections\]](#) Property

Debug Information	
Execute to the specified symbol after CPU Reset	Yes
Specified symbol	_main
Specify the debugged overlay section	Yes
Overlay sections	[3]
[0]	0x3000 - 0x301A
File	C:\Project\sample\DefaultBuild\sample.abs
Start address	HEX 3000
End address	HEX 301A
Priority section	ram01
[1]	0x3100 - 0x3110
[2]	0x3200 - 0x3203

This property shows the current information of overlay sections per address group in detail. Only [\[Priority section\]](#) can be changed.

- [\[File\]](#)

Name of the load module file in which the selected address group has been downloaded (this cannot be changed).

- [\[Start address\]](#)

First address of the selected address group (this cannot be changed).

- [\[End address\]](#)

Last address of the selected address group (this cannot be changed).

- [\[Priority section\]](#)

List of sections defined in the selected address group. You can choose a section to be debugged (priority section) from this list.

Cautions 1. Settings related to overlay sections are not saved in the project file.

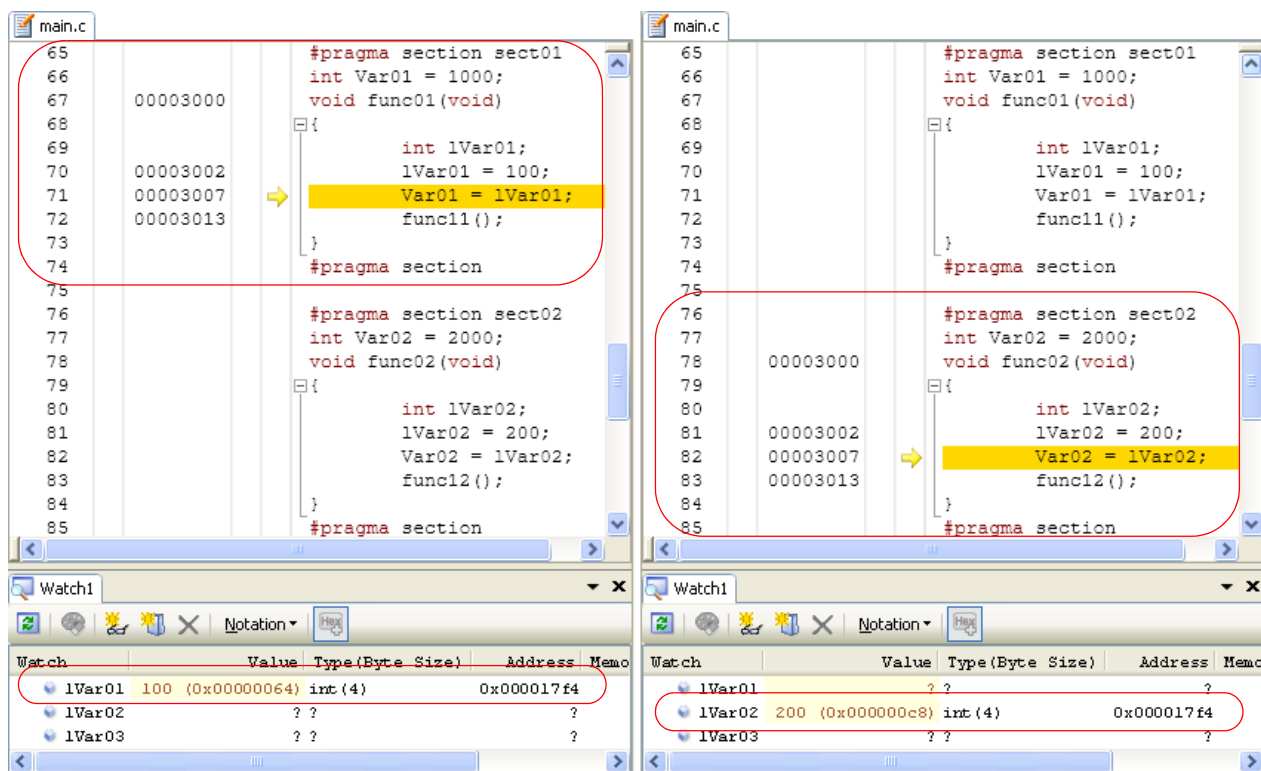
After you have downloaded a load module, select the priority section again.

2. Only debugging information is switched when the selection for [\[Priority section\]](#) is changed.

The debug tool does not copy data of the target section.

The following figure shows an example where "ram01" is selected as the priority section (left pane) and then changed to "ram02" (right pane) for the [Overlay sections] property in the case shown in Figure 2-79. Allocation of Overlay Sections.

Figure 2-82. Changing the Selection for [Priority section] from "ram01" to "ram02"



2.9 Execute Programs

This section describes how to execute programs.

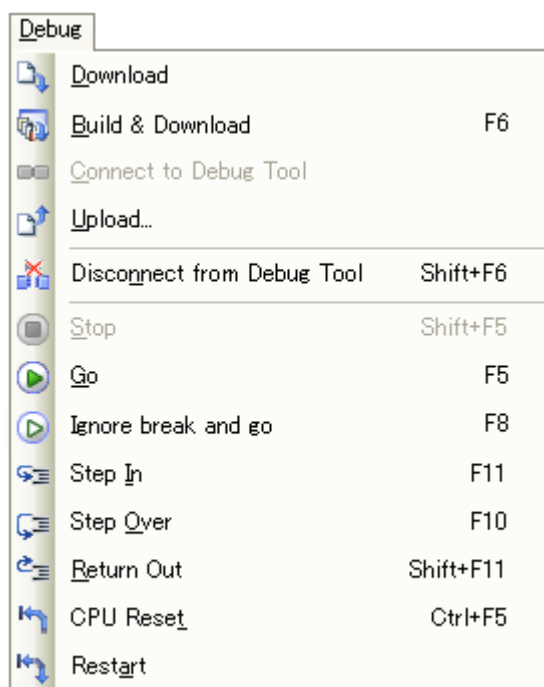
The major operations discussed in this section can be conducted either from the debug toolbar or the [Debug] menu in the [Main window](#), where commands to control the execution of programs are included.

Caution Items in the debug toolbar and the [Debug] menu will be disabled once the connection with the debug tool is lost.

Figure 2-83. Debug Toolbar



Figure 2-84. [Debug] Menu



2.9.1 Reset microcontroller (CPU)

To reset CPU, click the  button on the debug toolbar.

When CPU is reset, the current PC value is set to the reset address.

Remark You can automatically overwrite the value of I/O register/CPU register with the specified values after CPU reset (see ["2.18 Setting Up the Hook Process"](#) for details).

2.9.2 Execute programs

The following types of CubeSuite+ execution functions are provided.

Select any of the following operations according to the purpose of debugging.

See ["2.10 Stop Programs \(Break\)"](#) for details on how to stop the program in execution.


- (1) [Execute after resetting microcontroller \(CPU\)](#)
- (2) [Execute from the current address](#)

(3) Execute after changing PC value


Remark You can automatically overwrite the value of I/O register/CPU register with the specified values before executing the program (see "2.18 Setting Up the Hook Process" for details).

(1) Execute after resetting microcontroller (CPU)

You can reset the CPU and start the execution of the program from the reset address.

Click the  button on the debug toolbar.

When this operation is performed, the program continues to be executed until either of the following occurs:


- The  button has been clicked (see "2.10.1 Stop the program manually").
- The PC has reached a breakpoint (see "2.10.2 Stop the program at the arbitrary position (breakpoint)").
- A break event condition has been met (see "2.10.4 Stop the program with the access to variables/I/O registers").
- Other break factors have occurred.

Remark This operation is the same as when the  button is clicked after clicking the  button.


(2) Execute from the current address

Perform any of the following operations to start executing the program from the address at the current PC value.


(a) Normal execution

Click the  button on the debug toolbar.


When this operation is performed, the program continues to be executed until either of the following occurs:

- The  button has been clicked (see "2.10.1 Stop the program manually").
- The PC has reached a breakpoint (see "2.10.2 Stop the program at the arbitrary position (breakpoint)").
- A break event condition has been met (see "2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20]" and "2.10.4 Stop the program with the access to variables/I/O registers").
- Other break factors have occurred.

(b) Execution ignoring break-related events

Click the  button on the debug toolbar.

When this operation is performed, the program continues executing until either of the following occurs:


- The  button has been clicked (see "2.10.1 Stop the program manually").
- Other break factors have occurred.

Remark If you have started the execution with this operation, the occurrence of Printf event will also be ignored.

(c) Execution to the caret position

To start this operation, move the caret to the line/instruction at which you wish to stop the program in the [Editor panel/Disassemble panel](#), then select [Go to Here] from the context menu.

When this operation is performed, the program continues to be executed until either of the following occurs:

- The PC has reached the address of the caret position.
- The  button has been clicked (see "2.10.1 Stop the program manually").
- Other break factors have occurred.

Caution When the corresponding address of the line at the caret position does not exist, the program is executed to the corresponding address of the lower valid line (if the corresponding address does not exist, an error message will appear).

Remark If you have started the execution with this operation, the occurrence of Printf event will also be ignored.

(3) Execute after changing PC value

The program is executed after you have forcibly changed the current PC value to an arbitrary position.

To start this operation, move the caret to the line/instruction at which you wish to start the program in the [Editor panel/Disassemble panel](#), then select [Set PC to Here] from the context menu (the current PC value is set to the address of the line/instruction where the caret currently exists).

Then execute either one of the execution method described in "(2) [Execute from the current address](#)".

2.9.3 Execute programs in steps

When either of the following operation has occurred, the program will stop automatically after conducting step execution in the source level (1 line of source text) or in the instruction level (1 instruction).

Once the program is stopped, the contents of each panel will be updated automatically. As such, step execution is suited for debugging the program execution in transition either in source or instruction level.

The unit in which the program is step-executed is determined automatically depending on which panel is currently in focus:

- When the focus is not in the [Disassemble panel](#): Step execution^{Note} in the source level
- When the focus is in the [Disassemble panel](#): Step execution in the instruction level

Note If the line information does not exist in the address specified by the current PC value, the step execution is conducted in instruction level.

Step execution is divided into the following types:

- (1) [Step in function \(Step in execution\)](#)
- (2) [Step over function \(Step over execution\)](#)
- (3) [Execute until return is completed \(Return out execution\)](#)

Cautions 1. Break points, break events, and Printf events that have been set do not occur during step execution.

2. [E1] [E20]


Interrupts are not acknowledged during step execution.

[Simulator]

You may jump to an interrupt handler during step execution.

(1) Step in function (Step in execution)

When the function is called, the program is stopped at the top of the called function.


Click the  button on the debug toolbar to perform Step in execution.


Cautions 1. Step-in execution cannot be performed for a function that has no debug information.

2. If Step in execution is performed for the longjmp function, program execution may not complete and may wait for a time-out.

(2) Step over function (Step over execution)


In the case of a function call by a jump to subroutine instruction, all the source lines/instructions in the function are treated as one step and the program will be executed until reaching the position to which it returns from the function (step execution will continue until the same nest is formed as when a jump to subroutine instruction has been executed).

Click the  button on the debug toolbar to perform Step over execution.

In the case of other than a jump to subroutine instruction, operation is the same as when the  button is clicked.

Caution If Step over execution is performed for the longjmp function, execution processing may not complete and may wait for a time-out.

(3) Execute until return is completed (Return out execution)

Step-execute the program so that the program will stop when it returns from the current function to the caller function. When the execution of source line/instruction that require checking has been completed, you can perform step execution using this instruction so that you can make the program return to the caller function without step executing the remaining instructions inside the function. This instruction can be performed by clicking the  button on the debug toolbar.

- Cautions**
1. If a program is returned out in the main function, it will break inside the start-up routine.
 2. If a program is returned out in a function that has called the longjmp function, break may not occur.
 3. Executing return out from the recursive function may cause the program to run in a free-run mode.

2.9.4 Execute a specified routine [E1] [E20]

You can execute a specified routine immediately before the start or stop of the program execution. Setting an arbitrary routine allows you to control the target system in synchronization with the execution or stop of the program. See "2.10 Stop Programs (Break)" on how to stop a program in execution. This section describes the procedure for specifying a routine to be executed.

You can specify a routine in the [System] [E1] [E20] category on the [Debug Tool Settings] tab in the Property panel.

- When executing immediately before the program execution

Select [Yes] in the [Execute the specified routine immediately before execution of the user program] property, and enter the start address in the [The routine to run immediately before starting execution] property that is displayed. The specified routine will be executed immediately before the program execution.

- When executing immediately after the break

Select [Yes] in the [Execute the specified routine immediately after halting of the user program] property, and enter the start address in the [The routine to run immediately after halting execution] property that is displayed. The specified routine will be executed immediately after the stop of the program.

Figure 2-85. [System] category

System	
Debug the program re-writing the on-chip PROGRAM ROM	No
Debug the program re-writing the on-chip DATA FLASH	No
Execute the specified routine immediately before execution of the user program	Yes
Routine to run immediately before execution starts	1000
Execute the specified routine immediately after the user program stops	Yes
Routine to run immediately after execution stops	

- Cautions 1.** In the [The routine to run immediately before starting execution] property and the [The routine to run immediately after halting execution] property, you can also specify function name (when using C language) or label name (when using Assembly language) as property values.
2. Use an interrupt stack when using a stack inside the specified routine.
 3. Describe a return sub-routine (RTS) instruction for terminating the specified routine processing.
 4. The processing time of one specified routine must not exceed 100ms. If a clock remains halted in the specified routine, it may affect the control over the debug tool.
 5. The register value for the start of the specified routine is not determined. As such, you need to perform initial setting of the register value in the specified routine.
 6. Execution of a specified routine starts in the supervisor mode. Do not switch it to the user mode.
 7. When executing a specified routine, do not perform memory access/download/break point setting to the program area of the specified routine.
 8. When executing a specified routine, the 4 bytes indicated by the interrupt stack pointer will be used for the control on the debug tool side.
 9. Following restrictions will apply to the general registers and flags used in the specified routine.

ISP Register	Once the specified routine is executed, the value should be returned to the one at the start of the execution.
U Flag	Switching to the user mode is prohibited while the specified routine is in execution.
I Flag	Interrupt is prohibited while the specified routine is in execution.
PM Flag	Switching to the user mode is prohibited while the specified routine is in execution.

10. While the specified routine is in execution, its event settings will be disabled along with Trace, Break, Real-time RAM monitor and Timer functions.
11. Non-maskable interrupt is always prohibited while the specified routine is in execution.
12. When starting a program after executing the specified routine, the state of microcontroller will be as follows


General Register	Remains in the state at which the program stopped or reflects the user setting. The contents of the register following the execution of the specified routine are not reflected.
Memory	Reflects the memory access made after the execution of the specified routine.
Peripheral Functions	The operation of microcontroller peripheral functions after the execution of the specified routine continues.

2.10 Stop Programs (Break)

This section describes how to stop the program in execution.

Remark When the program in execution is stopped, a statement of the cause of the break appears on the [Status bar](#) in the [Main window](#).

2.10.1 Stop the program manually

The program in execution is forcibly stopped by clicking the  button on the debug toolbar.

2.10.2 Stop the program at the arbitrary position (breakpoint)

The program in execution can be stopped at the arbitrary position by setting a breakpoint. A breakpoint can be set with a single click of the mouse.

You need to configure the type of breakpoints to use before setting a breakpoint.

This section describes the following.

- (1) [Configure types/movements of breakpoints to use \[E1\] \[E20\]](#)
- (2) [Set a breakpoint](#)
- (3) [Edit a hardware breakpoint](#)
- (4) [Delete a breakpoint](#)

Caution [E1 [RX630, RX631, RX63N, and RX210 Groups]] [E20 [RX630, RX631, RX63N, and RX210 Groups]]
If a break occurs for an executed-related break that is set in the WAIT instruction or an instruction immediately preceding it, a time-out error may occur. To cause the target program to break in the WAIT instruction or an instruction immediately preceding it, use a breakpoint.

(1) Configure types/movements of breakpoints to use [E1] [E20]

Breakpoints to use are set in the [\[Break\] \[E1\] \[E20\]](#) category on the [\[Debug Tool Settings\] tab](#) of the [Property panel](#). The setting method differs depending on the debug tool used.

Figure 2-86. [Break] Category [E1][E20]



Specify the type of preferential breakpoint (breakpoint that can be set with a single click of the mouse) in the [\[First using type of breakpoint\]](#) property.

Note, however, that if the number of the set breakpoints of the specified type exceeds the limit settable (see "[\(1\) Limitation on the number of valid events](#)"), a breakpoint of another type will be used.

Select the breakpoint type from the drop-down list according to their functions.

Software break	Temporarily replaces instruction code for the specified address with break instruction and stops the program when this instruction is executed. Once set, it is handled as a software break event.
Hardware break	The debug tool consecutively checks the break condition while the program is in execution and stops the program when the condition is met (default) ^{Note} . Once set, it is handed as a hardware break event (execution type).

Note Hardware break is a "before-execution" break as the program will break before executing instruction at the specified address. This function is implemented using the debug tool resources.

(2) Set a breakpoint

Perform this operation in the [Editor panel/Disassemble panel](#) in which the source text/disassembly text is displayed.

In the panel's event area, click on the location where you want to set a breakpoint.

A breakpoint is set to the instruction at the start address corresponding to the clicked line.

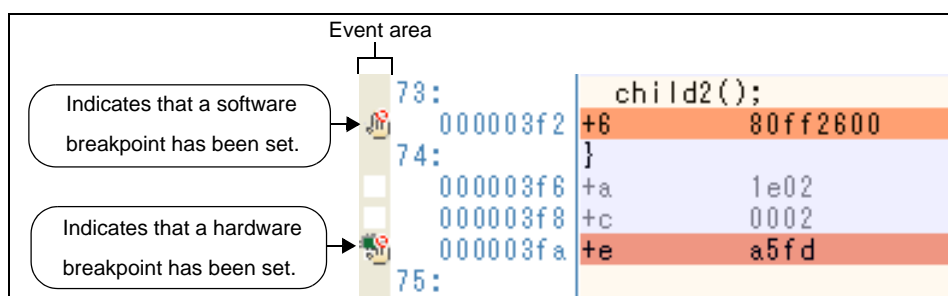
When a breakpoint is set, the following event mark appears at the breakpoint location, and the source text line/disassembled text line is highlighted.

It is interpreted as if a break event (Software or Hardware Break event) has been set at the target address, and it is managed in the [Events panel](#) (see "2.17 Event Management" for details).

Table 2-5. Event Marks of the Breakpoint

Event Type	Event Mark
Software Break ([E1] [E20])	
Hardware Break	

Figure 2-87. Breakpoint Setting Example




Cautions 1. Since a breakpoint is set as a break event and managed as an event, restrictions apply to the number of breakpoints that can be simultaneously set. Also see "2.17.7 Points to note regarding event setting" for details on breakpoints (e.g. limits on the number of valid events).

2. Breakpoints cannot be set to lines with no address indication.

3. [E1] [E20]

Software breakpoints replace the instructions at the corresponding addresses. They cannot be specified in the area other than the internal ROM area and internal RAM area.

Remarks 1. Event marks differ depending on the event state (see "2.17.1 Changing states of setting (Enabled/Disabled)").

When an event is set at a point for which another event is already set, the event mark () is displayed to indicate that more than one event is set at the point.

2. [Simulator]

The type of breakpoint that can be set is locked to hardware breakpoints.

3. [E1] [E20]

You can set hardware breakpoints/software breakpoints without depending on the specification of "(1) Configure types/movements of breakpoints to use [E1] [E20]" by following the step below.

Type	Operation1	Operation2
Hardware breakpoint	[Ctrl] + mouse click	Select [Break Settings] >> [Set Hardware Break] from the context menu.
Software breakpoint	[Shift] + mouse click	Select [Break Settings] >> [Set Software Break] from the context menu.


Caution Operation 1 is enabled only in the [Disassemble panel](#).

(3) Edit a hardware breakpoint

When editing a break point you have set, first open the [Events panel](#) by selecting [View] menu >> [Event]. After selecting the target hardware break point on the Event panel, click "Edit the conditions..." in the Context menu. This will open a dialog box in which you can edit the selected hardware break point. For details on editing in the dialog box, see "(1) [Editing execution-related events](#)".

(4) Delete a breakpoint

Click event marks displayed in the [Editor panel/Disassemble panel](#) to delete set breakpoints (the event mark will be erased).

You can also delete a break point on the [Events panel](#) which opens by selecting [View] menu >> [Event]. After selecting the target breakpoint on the panel, click () in the tool bar to delete it (see "2.17.5 [Deleting events](#)" for details).

2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20]

The program in execution can be stopped at the arbitrary position by setting a break event (execution type). It is an "after-execution" break as the program will break after executing instruction at the specified address. This function is implemented using the debug tool resources.

This section describes the following operations.

- (1) [Set a break event \(execution type\)](#)
- (2) [Edit a break event \(execution type\)](#)
- (3) [Delete a break event \(execution type\)](#)

(1) Set a break event (execution type)

Perform this operation in the [Editor panel/Disassemble panel](#) in which the source text/disassembly text is displayed.

After moving the caret to the target line, Select [Break Settings] >> [Set Combination Break].

A break event is set to the instruction at the start address corresponding to the clicked line. When a break event (execution type) is set, the event mark identical to that of hardware breakpoint appears, and the disassembled text line will be highlighted (see "(2) [Set a breakpoint](#)").

When you have performed this operation, it is interpreted as if a break event (execution type) has been set at the target address, and it is managed in the [Events panel](#) (see "2.17 [Event Management](#)" for details). It is registered as "after execution" in the detailed information on the combination break in the [Events panel](#).

Caution When setting a break event (execution type), also see "2.17.7 [Points to note regarding event setting](#)" for details (e.g. limits on the number of valid events).

Figure 2-88. Example of Setting a Break Event (Execution type) on a Line in the Disassemble Text [E1] [E20]

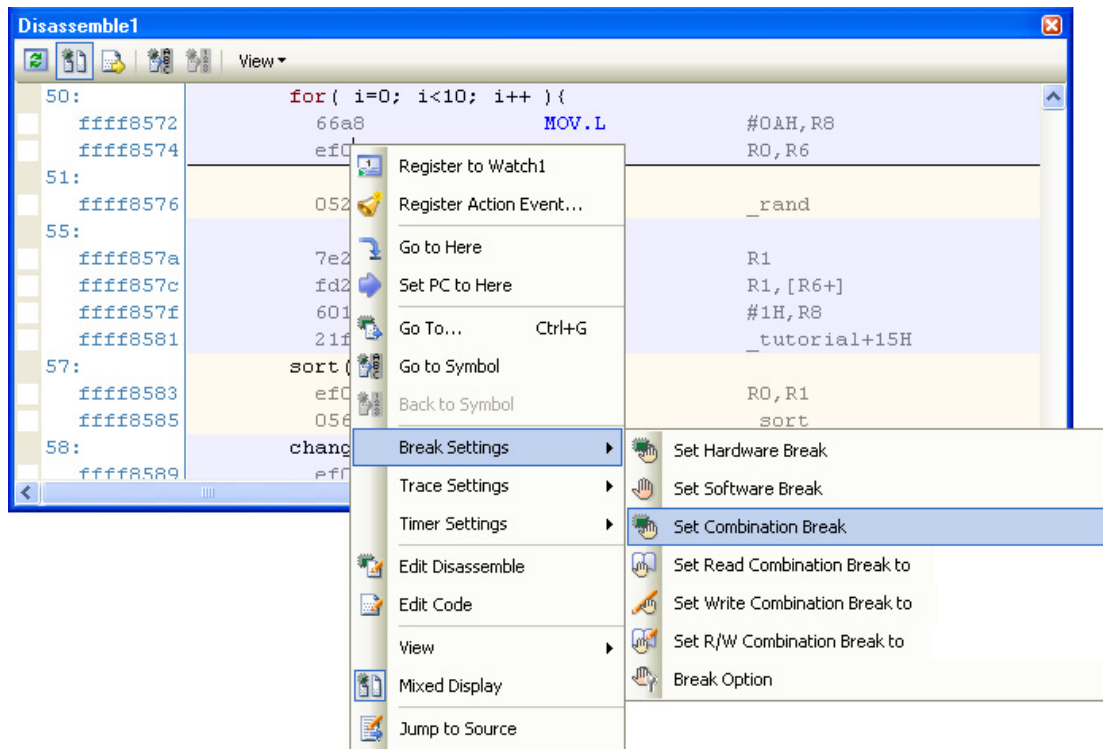
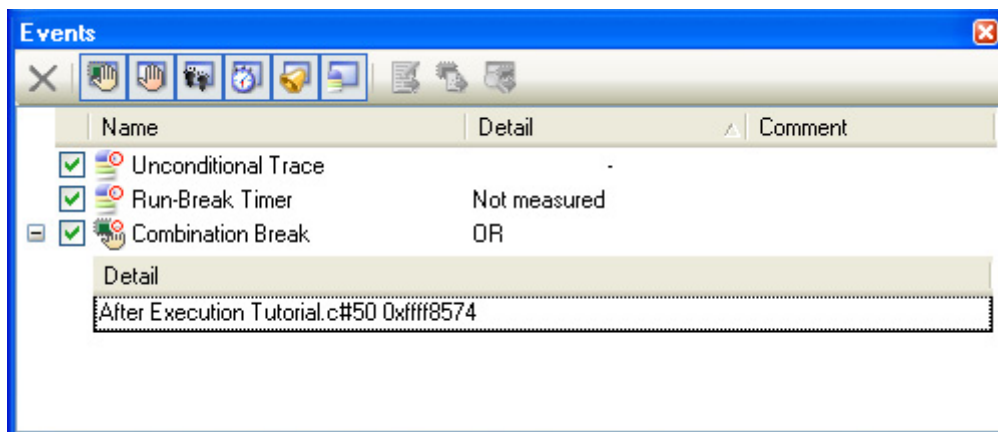



Figure 2-89. Example of Setting a Break Event (Execution type) in the Event Panel [E1] [E20]

**(2) Edit a break event (execution type)**

When editing a break event (execution type) you have set, first open the [Events panel](#) by selecting [View] menu >> [Event]. After selecting "after-execution" break displayed in the detailed information on the combined break in the Event panel, click [Edit Condition ...] in the Context menu. This will open a dialog box in which you can edit the selected break event. For details on editing in the dialog box, see "(1) [Editing execution-related events](#)".

(3) Delete a break event (execution type)

When deleting a break event (execution type) you have set, first open the [Events panel](#) by selecting [View] menu >> [Event]. After selecting the target combined break, click () button on the toolbar to delete it (see "[2.17.5 Deleting events](#)").

Each break event (execution type) can also be deleted by clicking on the event mark on the source or disassembled text.

Caution In the [Events panel](#), you cannot delete a break event (execution type) selectively in the combination breaks. All the break events (including access type) displayed in the detailed information on the combination break will be deleted.

2.10.4 Stop the program with the access to variables/I/O registers

By setting a break event (access type) with the access, the program can be stopped when an arbitrary variable or I/O register is accessed with the specified type.

You can also limit the accessed value.

The following types can be specified with the access.

Table 2-6. Types of Accesses to Variables

Access Type	Description
Read	The program is stopped with the read access to (after reading) the specified variable/I/O register.
Write	The program is stopped with the write access to (after writing) the specified variable/I/O register.
Read/Write	The program is stopped with the read access/write access to (after reading or writing) the specified variable/I/O register.

Cautions 1. The program is not stopped with the access via DMAC (Dynamic Memory Access Controller) / DTC (Data Transfer Controller).

2. [Simulator]

With the string operation instruction and multiply-accumulation instruction, only the final data access is subjected to the event check.

This section describes the following.

- (1) [Set a break event \(access type\) to a variable/I/O register](#)
- (2) [Edit a break event \(access type\) to a variable/I/O register](#)
- (3) [Delete a break event \(access type\) to a variable/I/O register](#)

(1) Set a break event (access type) to a variable/I/O register

Use one of the following methods to set a break event (access type) that stops programs with the access to a variable/I/O register.

Cautions 1. Also see "[2.17.7 Points to note regarding event setting](#)" for details on break event (access type) settings, including the allowable number of valid events.

2. [E1] [E20]

Break events (access type) are displayed as Read, Write, or Read/Write as part of detailed information on combination breaks in the [Events panel](#).

(a) Set a break event (access type) to a variable/I/O register in the source text/disassembled text

Perform this operation in the [Editor panel/Disassemble panel](#) in which the source text/disassembly text is displayed.

Follow the operation listed below from the context menu after selecting an arbitrary variable or I/O register in the source text/disassembled text. Note, however, that only global variables, static variables inside functions, and file-internal static variables can be used.

By performing the following operation, it is interpreted as if a Break event (access type) has been set at the target variable/I/O register, and it is managed in the [Events panel](#) (see "[2.17 Event Management](#)" for details).

Access Type	Operation
Read	<p>[E1] [E20] Select [Break Settings] >> [Set Read Combination Break to], and then press the [Enter] key.</p> <p>[Simulator] Select [Break Settings] >> [Set Read Break to], and then press the [Enter] key.</p> <p>If you have specified a value in the text box in the menu, break will occur only when the specified value is used for the reading. On the other hand, if no value is specified, reading the selected variable by any value will cause the break to occur.</p>
Write	<p>[E1] [E20] Select [Break Settings] >> [Set Write Combination Break to], and then press the [Enter] key.</p> <p>[Simulator] Select [Break Settings] >> [Set Write Break to], and then press the [Enter] key.</p> <p>If you have specified a value in the text box in the menu, break will occur only when the specified value is used for the writing. On the other hand, if no value is specified, writing the selected variable by any value will cause the break to occur.</p>
Read/Write	<p>[E1] [E20] Select [Break Settings] >> [Set R/W Combination Break to], and then press the [Enter] key.</p> <p>[Simulator] Select [Break Settings] >> [Set R/W Break to], and then press the [Enter] key.</p> <p>If you have specified a value in the text box in the menu, break will occur only when the specified value is used for the reading/ writing. On the other hand, if no value is specified, reading/ writing the selected variable by any value will cause the break to occur.</p>

- Cautions**
- Variables within the current scope can be specified.**
 - Variables or I/O register at lines that have no valid addresses cannot be used for break events.**

Figure 2-90. Example of Setting a Break Event (Access type) on Variable in the Source Text [E1] [E20]

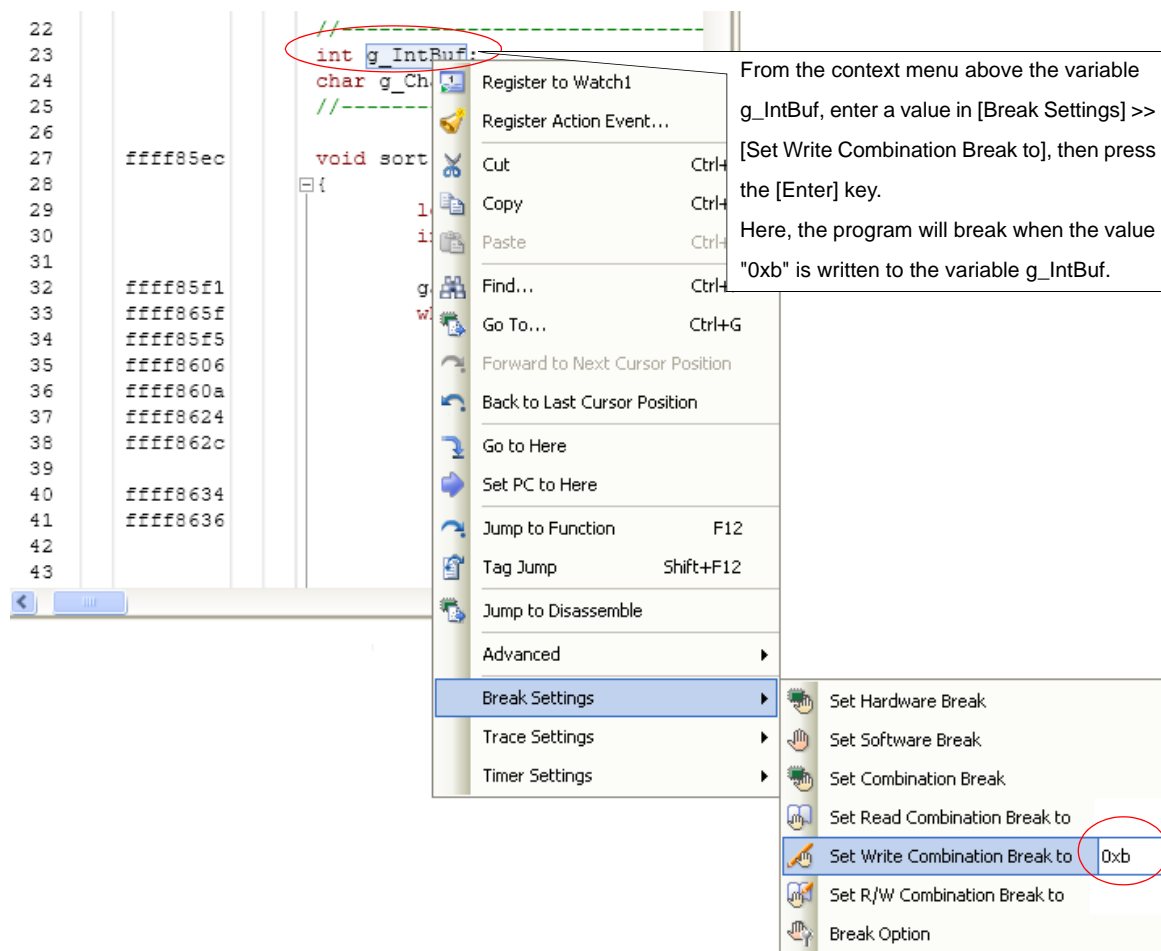


Figure 2-91. Example of Setting a Break Event (Access type) in the Event Panel [E1] [E20]

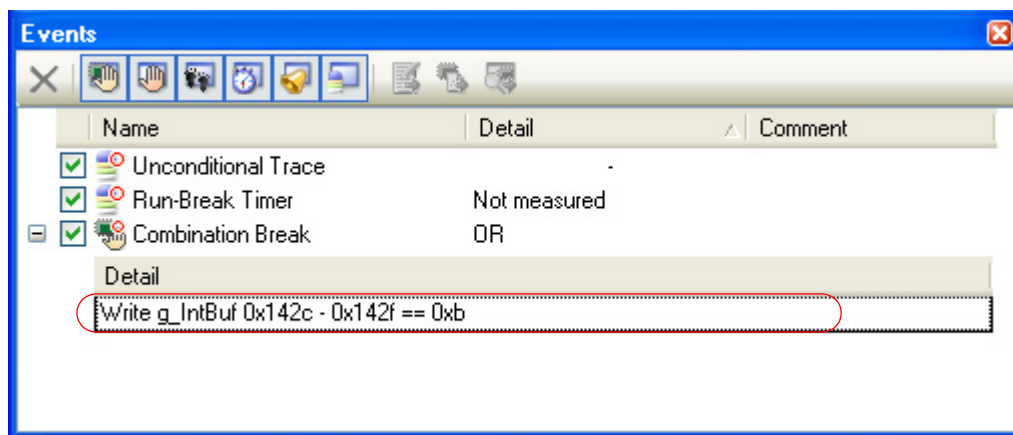


Figure 2-92. Example of Setting a Break Event (Access type) on a Variable in the Source Text [Simulator]

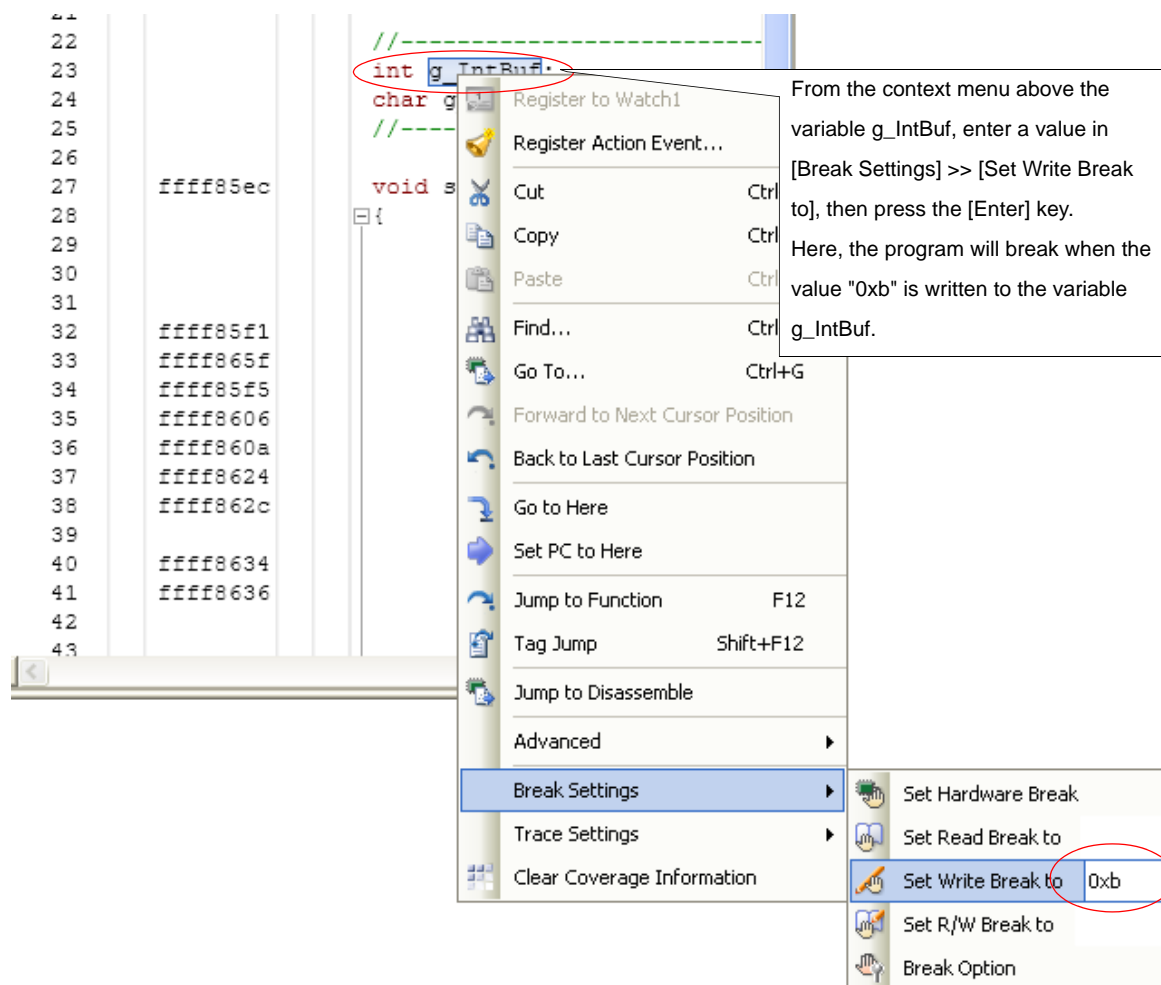
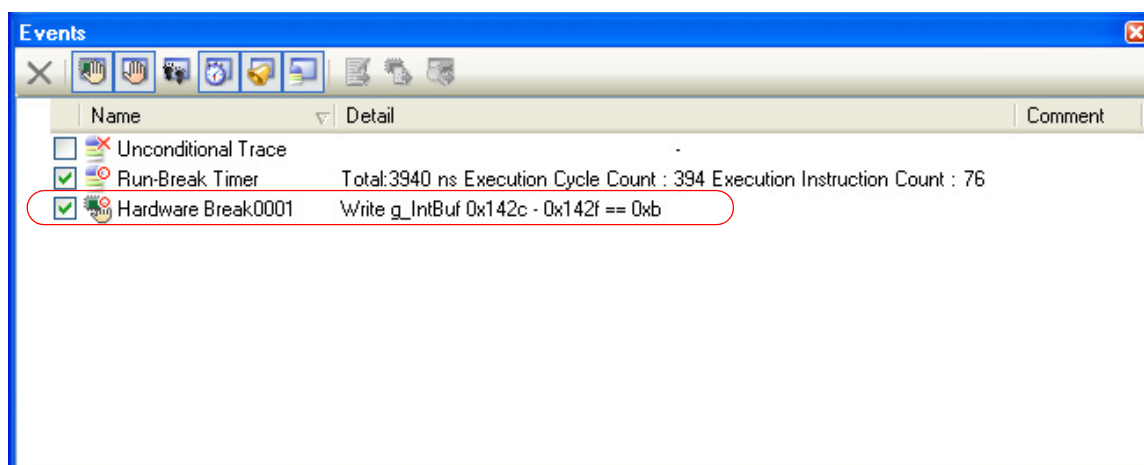


Figure 2-93. Example of Setting a Break Event (Access type) in the Event Panel [Simulator]

**(b) Set a break event (access type) to a registered watch-expression**

You can set break events in the [Watch panel](#).

Follow the operation listed below from the context menu after selecting the registered watch-expression (multiple selections not allowed).

Note, however, that only global variables, static variables inside functions, file-internal static variables, and I/O register can be used.

By performing the following operation, it is interpreted as if a Break event (access type) has been set at the target watch-expression, and it is managed in the [Events panel](#) (see "2.17 Event Management" for details).

Access Type	Operation
Read	<p>[E1] [E20]</p> <p>Select [Access Break] >> [Set Read Combination Break to], and then press the [Enter] key.</p> <p>[Simulator]</p> <p>Select [Access Break] >> [Set Read Break to], and then press the [Enter] key.</p> <p>If you have specified a value in the text box in the menu, break will occur only when the specified value is used for the reading. On the other hand, if no value is specified, reading the selected watch-expression by any value will cause the break to occur.</p>
Write	<p>[E1] [E20]</p> <p>Select [Access Break] >> [Set Write Combination Break to], and then press the [Enter] key.</p> <p>[Simulator]</p> <p>Select [Access Break] >> [Set Write Break to], and then press the [Enter] key.</p> <p>If you have specified a value in the text box in the menu, break will occur only when the specified value is used for the writing. On the other hand, if no value is specified, writing the selected watch-expression by any value will cause the break to occur.</p>
Read/Write	<p>[E1] [E20]</p> <p>Select [Access Break] >> [Set R/W Combination Break to], and then press the [Enter] key.</p> <p>[Simulator]</p> <p>Select [Access Break] >> [Set R/W Break to], and then press the [Enter] key.</p> <p>If you have specified a value in the text box in the menu, break will occur only when the specified value is used for the reading/ writing. On the other hand, if no value is specified, reading/ writing the selected watch-expression by any value will cause the break to occur.</p>

Remark A watch-expression within the current scope can be specified.

To target a watch-expression outside the current scope, select a watch-expression with a specified scope.

Figure 2-94. Example of Setting a Break Event (Access type) on a Watch-Expression [E1] [E20]

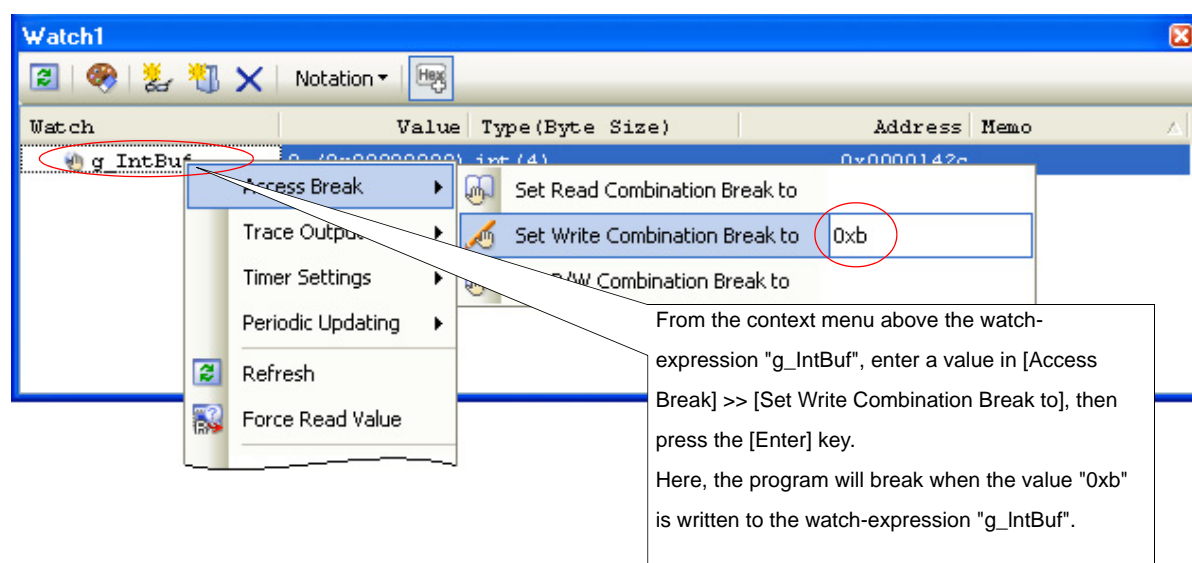
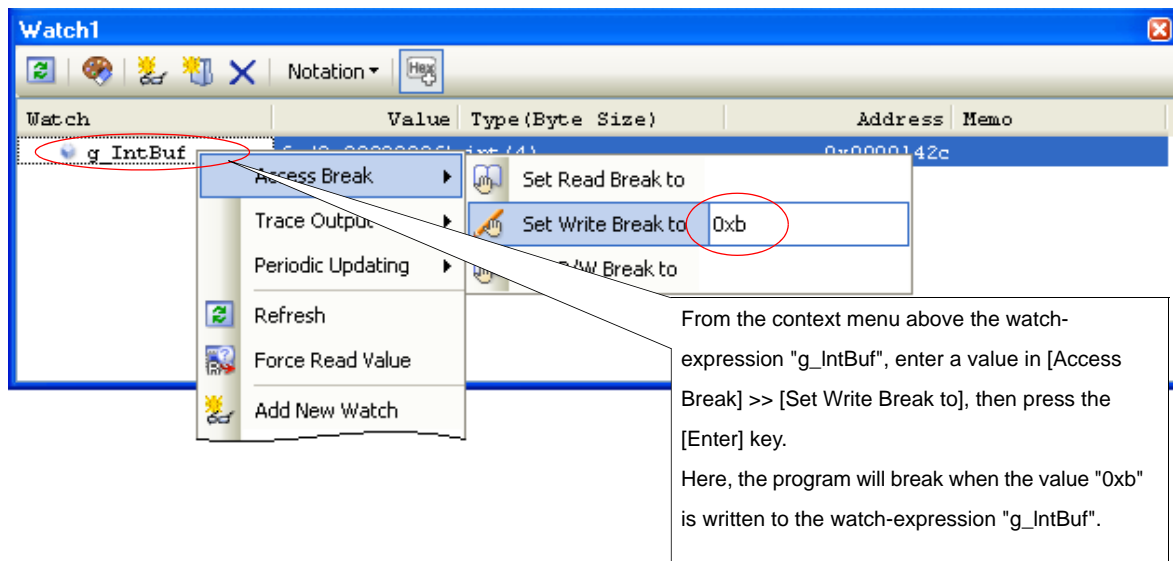


Figure 2-95. Example of Setting a Break Event (Access type) on a Watch-Expression [Simulator]

**(2) Edit a break event (access type) to a variable/IO register**

When editing a break event (access type) you have set, first open the [Events panel](#) by selecting [View] menu >> [Event]. After selecting the target break event (access type) in the [Events panel](#), click [Edit Condition ...] in the Context menu. This will open a dialog box in which you can edit the selected break event. For details on editing in the dialog box, see "(2) [Editing access-related events](#)".

(3) Delete a break event (access type) to a variable/IO register

When deleting a break event (access type) you have set, first open the [Events panel](#) by selecting [View] menu >> [Event], and then perform the following operations. Note that event name differs depending on the debug tool to use.

(a) [E1] [E20]

After selecting the combination break you wish to delete, click () button on the toolbar in the above panel (see "2.17.5 [Deleting events](#)").

Each break event (access type) can also be deleted by clicking on the event mark on the disassembled text.

Caution In the [Events panel](#), you cannot delete a break event (access type) selectively in the combination breaks. All the break events (including execution type) displayed in the detailed information on the combination break will be deleted.

(b) [Simulator]

After selecting the hardware break (access type) you wish to delete, click () button on the toolbar in the above panel (see "2.17.5 [Deleting events](#)").

2.10.5 Set multiple break events in combination (Combination break) [E1] [E20]

By setting more than one execution-type break event and access-type break event in combination, you can stop the program when the combination condition is satisfied by the set break event.

Only one combination break can be set. Therefore, when two or more break events have been set, break event will be added consecutively to the detailed information of one combination break in the [Events panel](#). Combination conditions listed below can be specified for the combination break.

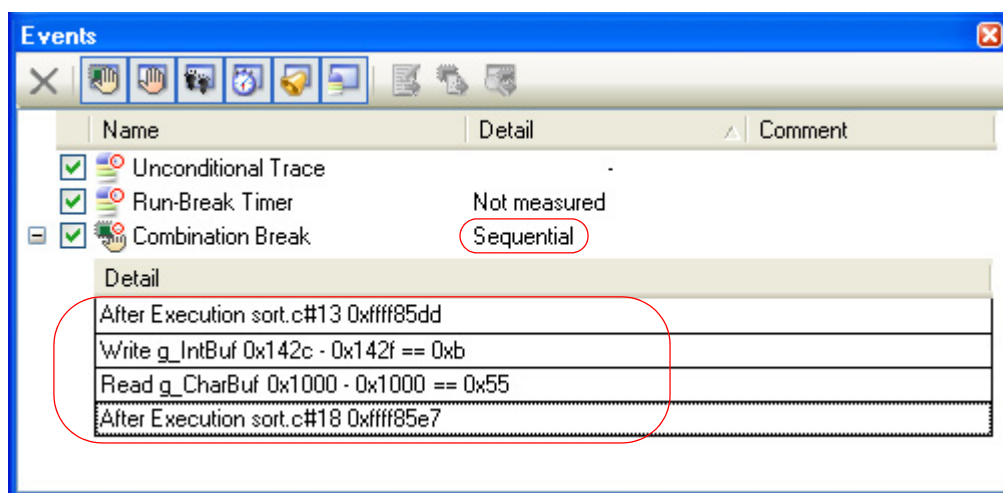
Table 2-7. Combination conditions for the combination break

Combination conditions	Description
OR	Break will occur when any one of the break event conditions is satisfied.
AND	Break will occur when all the break event conditions are satisfied irrespective of the time line.
Sequential	Break will occur when the break event conditions are satisfied in the specified sequence. Only one break event in the combination breaks can be registered as a reset event (R event). When the registered break event is encountered, all the other break event conditions which have been holding until then will be cleared.

When editing a combination break, select the combination break in the [Events panel](#) and click [Edit Condition ...] in the Context menu. This will open a dialog box in which you can edit the combination break. For details on editing in the dialog box, see "(3) [Editing combination conditions of events \[E1\] \[E20\]](#)".

- Cautions 1.** For the combined break settings, also see "[2.17.7 Points to note regarding event setting](#)" (e.g. limits on the number of valid events).
- When the combination condition is "Sequential", break event (access type) can be specified for the 1st to the 3rd position.
 - When the combination condition is "Sequential", break event (access type) for which address range condition is set can be specified only for the 1st position.

Figure 2-96. Example of Setting a Combination Break (Sequential) in the Event Panel [E1] [E20]



2.10.6 Other break factors

Other than the causes described above, a program can be stopped by the following break factors.

You can check such break factors on the [Status bar](#) in the [Main window](#).

Table 2-8. Other break factors

Break factors	Debug Tool to Use	
	E1(Serial)/E1(JTAG)/ E20(Serial)/E20(JTAG)	Simulator
Trace memory full ^{Note 1}	✓	✓
Temporary break occurred	✓	✓

Break factors	Debug Tool to Use	
	E1(Serial)/E1(JTAG)/ E20(Serial)/E20(JTAG)	Simulator
Execution failed or cause unknown	✓	-
WAIT Instruction executed	-	✓ Note 3
Undefined instruction exception encountered	-	✓ Note 3
Privileged instruction exception encountered	-	✓ Note 3
Access exception encountered	-	✓ Note 3
Floating-point exception encountered Note 2	-	✓ Note 3
Interrupt encountered	-	✓ Note 3
INT instruction exception encountered	-	✓ Note 3
BRK instruction exception encountered	-	✓ Note 3
Peripheral function simulation error occurred	-	✓
Illegal memory access made	-	✓ Note 3
Stream input/output error occurred	-	✓
Allocation of coverage memory failed	-	✓
Allocation of trace memory failed	-	✓

- Notes 1.** The operation depends on the setting of the [Operation after trace memory is full] property in the [Trace] category on the [Debug Tool Settings] tab of the Property panel.
- 2.** Applicable only to the RX600 Series.
- 3. [Simulator]**
The operation depends on the settings of each property in the [Execution Mode] [Simulator] category on the [Debug Tool Settings] tab of the Property panel.

2.11 Displaying and Changing Memory, Registers, and Variables

This section describes how to display or change the contents of memory, registers, and variables.

2.11.1 Displaying and changing memory contents

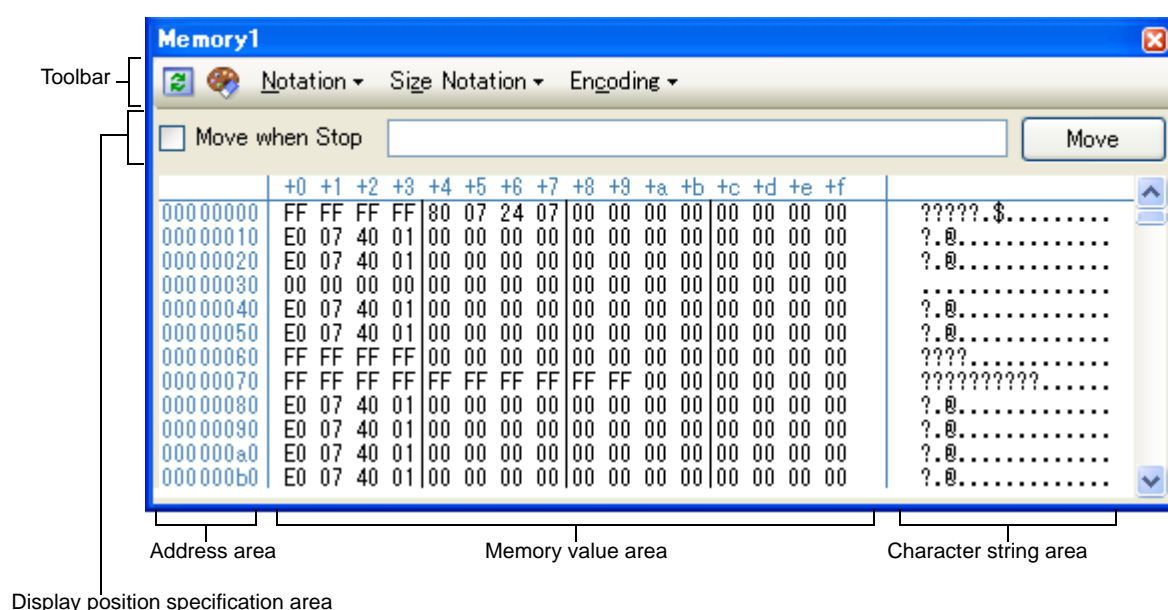
Use the [Memory panel](#) shown below to display memory contents or change values.


Choose [Memory] from the [View] menu and then [Memory1 - 4].

Up to four pieces of [Memory panels](#) can be opened at a time. Each panel is discriminated by the name "Memory1," "Memory2," "Memory 3," and "Memory 4" in the title bar.

For details on how to read each area and details about their functionality, see the section where the [Memory panel](#) is described.

Figure 2-97. Displaying Memory Contents



Remark In the [Scroll Range Settings dialog box](#) which opens by clicking on [View] >>  button in the toolbar, you can set the scroll range of the vertical scroll bar on this panel.

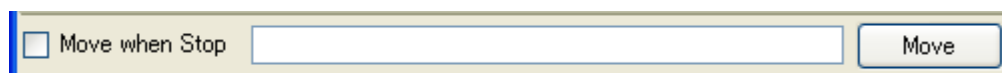
This section describes the following.

- (1) [Specifying the display position](#)
- (2) [Changing the display form of values](#)
- (3) [Changing memory contents](#)
- (4) [Displaying and changing memory contents during program execution](#)
- (5) [Searching for memory contents](#)
- (6) [Collectively changing \(initializing\) memory contents](#)
- (7) [Saving displayed memory contents](#)

(1) Specifying the display position

By specifying an address expression in the display position specification area, it is possible to specify the position at which CubeSuite+ starts displaying memory values. (CubeSuite+ starts displaying, by default, from the address "0.")

Figure 2-98. Display Position Specification Area (Memory Panel)

**(a) Specifying an address expression**

Directly enter in the text box an address expression that designates the address of the memory value you want to be displayed. An input expression of up to 1,024 characters can be specified, the calculation result of which is handled as the address for the display start position.

However, no address expressions can be specified that exceed the address space of the microcontroller.

- Remarks 1.** By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)")
- 2.** If the specified address expression is the symbol and its size can be recognized, everything from the start address to the end address of that symbol is displayed selected.

(b) Specifying whether the address expression is evaluated automatically or manually
















The timing with which the display start position will be changed is determined by selection of the [Move When Halted] checkbox and by the [Move] button.

[Move When Halted]	<input checked="" type="checkbox"/>	The address expression is automatically evaluated after the program has halted and the caret moves to the address derived from that calculation.
	<input type="checkbox"/>	The address expression is not automatically evaluated after the program has halted. In this case, the address expression is evaluated by clicking the [Move] button.
[Move]		If the [Move When Halted] checkbox is not checked, the address expression is evaluated by clicking this button and the caret moves to the address derived from that calculation.

(2) Changing the display form of values

The forms in which data are displayed in the memory value area and character string area can be changed freely by using the toolbar buttons shown below.

However, these buttons are disabled during program execution.

Notation	Shows the following buttons that change the form in which memory values are displayed.
	Displays memory values in hexadecimal (default).
	Displays memory values in signed decimal.
	Displays memory values in unsigned decimal.
	Displays memory values in octal.
	Displays memory values in binary.
Size notation	Shows the following buttons that change the form in which size of memory values are displayed.
	Displays memory values in 4-bit width.
	Displays memory values in 8-bit width (default).
	Displays memory values in 16-bit width. Values are converted depending on the endian of the target memory area.
	Displays memory values in 32-bit width. Values are converted depending on the endian of the target memory area.
	Displays memory values in 64-bit width. Values are converted depending on the endian of the target memory area.
Encode	Shows the following buttons that change the encode in which character strings are displayed.
	Displays character strings in ASCII code (default).
	Displays character strings in Shift_JIS code.
	Displays character strings in EUC-JP code.
	Displays character strings in UTF-8 code.
	Displays character strings in UTF-16 code.

(3) Changing memory contents

Memory values can be edited.

In the memory value area or character string area, move the caret to the target memory value and then edit it directly from the keyboard.

When a memory value is edited, the altered portion is displayed in a different color. While in this state, press the [Enter] key and the altered value will be written into the target memory. (Pressing the [Esc] key before you hit the [Enter] key cancels editing.)

However, the character strings that can be entered when changing memory contents are limited to only those that are handleable by the currently specified system of notation. Also, changes in the character string area are only possible when character encode in "ASCII" is specified.

You can edit memory values even when the program is in execution. For details, see "(4) Displaying and changing memory contents during program execution".

In the case shown below, caution is required when editing the memory values.

- Examples 1.** When the maximum value of the displayed bit width is exceeded
If, while memory is displayed in decimal 8 bits, you edit the displayed value "105" by entering "3" for "1," then the altered value becomes "127" which is the maximum value.
- 2.** When "-" is entered in the middle of a numeric value
If, while memory is displayed in signed decimal 16 bits, you edit the displayed value "32768" by entering "-" in the middle of it like "32-68," then numerals "3" and "2" change to spaces and the altered value becomes "-68."

3. When a space character is entered in the middle of a numeric value
If, while memory is displayed in decimal 16 bits, you edit the displayed value "32767" by entering a space in the middle of it like "32 67," then numerals "3" and "2" change to spaces and the altered value becomes "67."
4. When the same value is entered
Even when you enter the same value as the current memory value, the specified value is written into memory.

(4) Displaying and changing memory contents during program execution

The [Memory panel](#) and [Watch panel](#) come with a realtime display update function that permits you to update display of, or even rewrite, the contents of memory or watch expressions in real time.

By enabling this realtime display update function, it is possible to display or change the values of memory or watch expressions even while the program is running, not just when the program is halted.

Cautions 1. [E20(JTAG)[RX600 Series]]

The trace function and the realtime RAM monitor function (RRM function) in part are usable exclusively of each other. Therefore, before making the settings below, you must finish the following settings on the [\[Debug Tool Settings\]](#) tab of the [Property panel](#) in order to specify the function to be used preferentially.

- [\[Trace\]](#) category >> [\[Usage of trace function\]](#) properties >> [\[Realtime RAM Monitor\]](#)

2. [E20(JTAG)[RX600 Series]]

If a reset such as a pin reset or watchdog timer reset is issued while a user program is running, the results of the real-time RAM monitor cannot be guaranteed (values displayed may be incorrect).

Remark The real-time display update function is materialized by the debug tool's RRM (Real-time RAM Monitor), pseudo-RRM, and DMM (Dynamic Memory Modification) functions.
The pseudo-RRM function temporarily momentarily breaks program execution to perform reads/writes by means of software emulation.

The subject area to and from which you can write and read using the realtime display update function varies depending on how the debug tool you use and the [Property panel](#) are set.

Referring to the set contents of respective properties in the [\[Access Memory While Running\]](#) and [\[Trace\]](#) categories on the [\[Debug Tool Settings\]](#) tab of the [Property panel](#) shown below, make settings in [Table 2-9.](#) through [Table 2-11.](#) according to the purpose of use of the realtime display update function.

Mark	Properties in [Trace] Category	Setting Value	Note
A	Usage of trace function	[Real-time RAM monitor]	E20(JTAG) [RX600 Series]

Mark	Properties in [Access Memory While Running] Category	Setting Value	Note
B	Access by stopping execution	[No] (default)	RRM/DMM functions enabled.
	Enable the automatic update of realtime display	[Yes] (default)	E20(JTAG) [RX600 Series]
C	Access by stopping execution	[Yes]	Pseudo RRM function enables.
D	Update display during the execution	[Yes] (default)	
	Display update interval[ms]	[Integer number between 100 and 65500]	

(a) For Read

Table 2-9. Subject Area of the Real-Time Display Update Function (Read by Pseudo-RRM Function)

Area	E1(Serial)/E1(JTAG)/E20(Serial)/ E20(JTAG)	
	Setting	Subject
Internal ROM	C+D	Entire range
Internal RAM		
Data flash		
Target memory		
Emulation memory	-	
CPU register	Not available	
I/O register		

Table 2-10. Subject Area of the Real-Time Display Update Function (Read by RRM Function)

Area	E20(JTAG) [RX600 Series]		Simulator	
	Setting	Subject	Setting	Subject
Internal ROM	A+B+D	Automatic setting area Note	D	Entire range
Internal RAM			-	
Data flash				
Target memory				
Emulation memory	-		D	Entire range
CPU register	Not available		-	
I/O register			D	Entire range

Note [E20(JTAG) [RX600 Series]]

The subject area of the RRM function is limited to a maximum of 4,096 bytes (4 areas each containing maximum 1024 bytes).

Therefore, CubeSuite+ automatically determines the subject of realtime display updates within the above limit according to the determination rules (order of priority) shown below. (Only the panel that is displayed in front of others immediately before program execution becomes the subject.)

(1) Watch expressions displayed on [Watch panel](#) are set in order from the top. (If multiple [Watch panels](#) are open, they are set in increasing order of panel number.)

(2) Memory contents displayed on [Memory panel](#) are set in increasing order of address. (If multiple [Memory panel](#) are open, they are set in increasing order of panel number.)

Note that if the subject area includes a read-prohibited area, display of that area is not updated.

(b) For write

Table 2-11. Subject Area of the Real-Time Display Update Function (Write by DMM Function)

Area	E1(Serial)/E1(JTAG)/E20(Serial)/ E20(JTAG)		Simulator	
	Setting	Subject	Setting	Subject
Internal ROM	Not available		D	Entire range
Internal RAM	C+D	Entire range		
Data flash	Not available		-	
Target memory	C+D	Entire range		
Emulation memory	-		D	Entire range
CPU register	Impossible		-	
I/O register			D	Entire range

Caution Local variables are not the subject of the real-time display update function.

Remark For details on how to rewrite values on [Memory panel](#) and [Watch panel](#), see "(3) Changing memory contents" and "(6) Changing the contents of watch expressions".

The memory values/watch expressions updated by the pseudo-RRM function are highlighted in pink on the [Memory panel](#), whereas those updated by the RRM function are highlighted as follows in accordance with the access status.

(The font and background colors depend on the configuration in the [\[General - Font and Color\]](#) category of the [Option dialog box](#).

Access Condition	Display Example
Read or fetch	00 00 00 00
Write	00 00 00 00
Read and write	00 00 00 00
Lost	00 00 00 00

Note The watch expression value will be displayed as "?" in the event of "Lost" on the [Watch panel](#).

Figure 2-99. Example of Memory Display by the Pseudo-RRM Function (Memory Panel) [E1][E20]

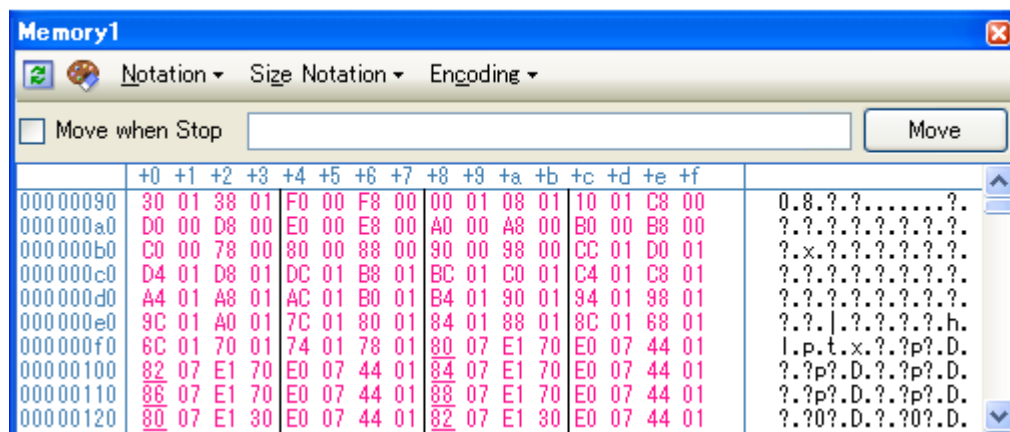


Figure 2-100. Example of Memory Display by the RRM Function (Memory Panel) [Simulator]

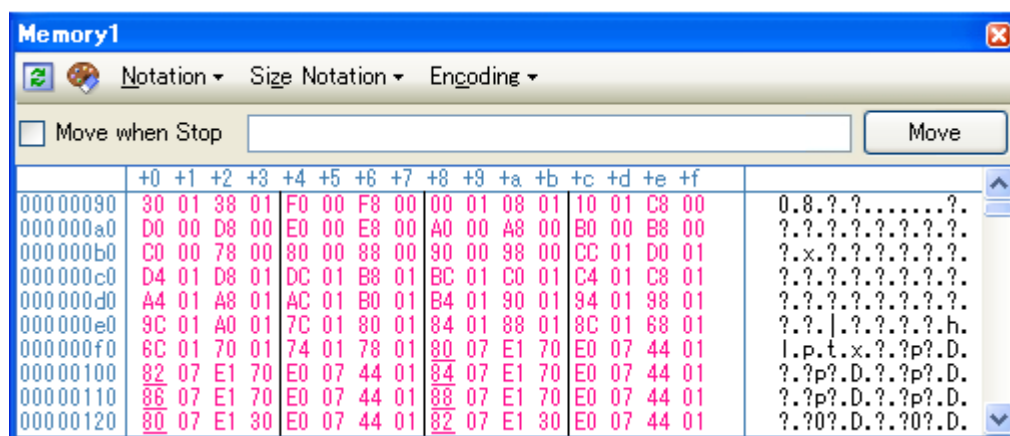
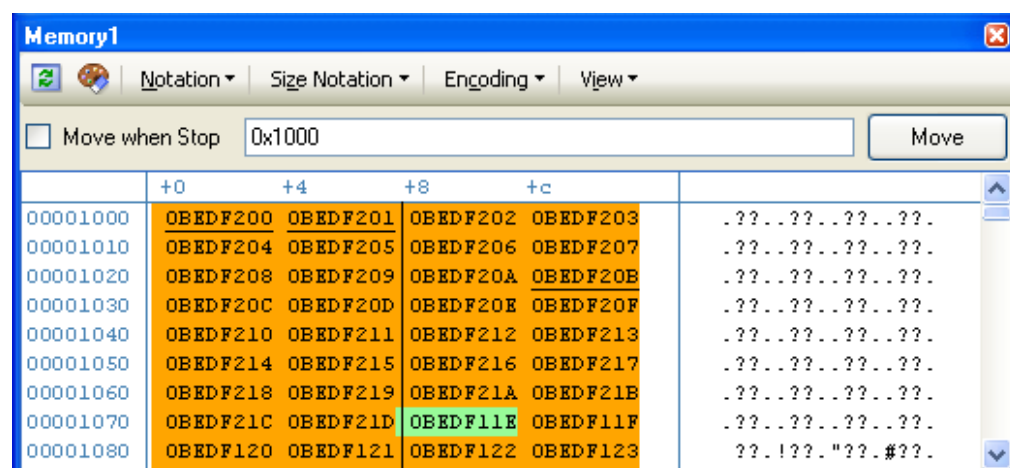


Figure 2-101. Example of Memory Display by the RRM Function (Memory Panel) [E20 (JTAG) [RX600 Series]

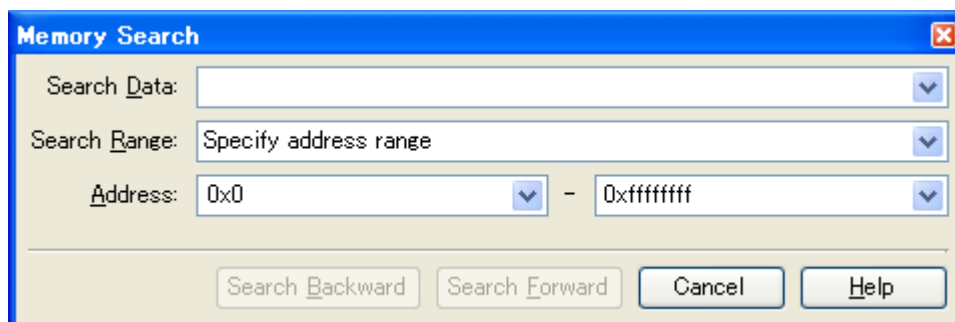


(5) Searching for memory contents

To search for memory values, use the [Memory Search dialog box](#) that is opened by selecting [Find...] on context menu. A search is conducted in the subject area, either the memory value or the character string area, where the caret exists.

In this dialog box, follow the procedure below to search for memory contents.

Figure 2-102. Searching for Memory Contents (Memory Search Dialog Box)



Caution Memory contents cannot be searched during program execution.

(a) Specify the [Search Data]

Specify the data you want to search.

Directly enter it in the text box (specifiable in up to 256 bytes) or select an input history item from the drop-down list (up to 10 history entries).

To perform a search in the memory value area, you need to enter data in the same display form (numeral system and size) as that area.

Also, if you perform a search in the character string area, you need to specify a character string as the search data. The specified character string, before being searched, is converted to data in appropriate encode form in which data are currently displayed in that area.

Note that if you select any memory value immediately before opening this dialog box, the value you've selected is displayed by default.

(b) Specifying the [Search Range]

Select the range in which you want to be searched from the drop-down list below.

Specify address range	A search is conducted within the address range specified by [Address].
Memory mapping	<p>A search is conducted within the selected range of memory mapping.</p> <p>This list item displays memory mappings individually (except non-mapped areas) that are displayed in the Memory Mapping dialog box [Simulator].</p> <p>Display form: <memory type> <address range> <size></p>

(c) Specifying the [Address]

This item is valid only when you've selected [Specify Address Range] in "(b) Specifying the [Search Range]."

Specify the "start address - end address" to set the address range in which you want a memory value to be searched. Directly enter address expressions in the respective text boxes (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries).

The calculation results of the address expressions you've entered are respectively handled as the start address and end address.

However, searchable addresses are limited to the upper-limit address of the program space (0xFFFFFFFF).

Also, no address values can be specified that are greater than the value representable in 32 bits.

- Remarks 1.** By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 Symbol name completion function").
- 2.** If the "start address" text box is blank, address "0x0" is assumed.

3. If the "end address" text box is blank, the upper-limit address of the microcontroller's address space is assumed.

(d) Clicking the [Search Backward] and [Search Forward] buttons

If you click the [Search Backward] button, a search is conducted in the direction toward smaller addresses within the specified range and the searched spot shown on [Memory panel](#) is in a selected state.

If you click the [Search Forward] button, a search is conducted in the direction toward larger addresses within the specified range and the searched spot shown on [Memory panel](#) is in a selected state.

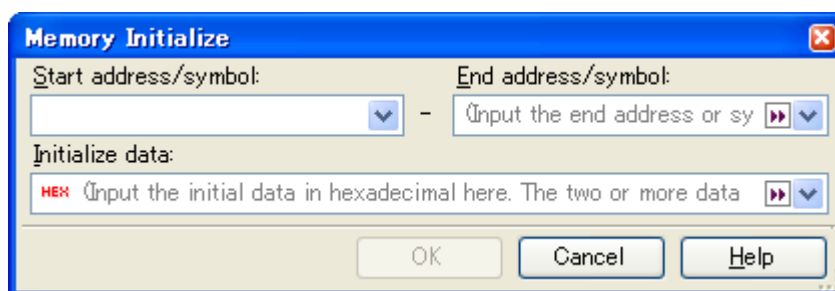
(6) Collectively changing (initializing) memory contents

Memory values can be changed collectively (initialized).

Selecting [Fill...] on context menu opens the [Memory Initialize dialog box](#) that permits you to change memory values in a specified address range collectively.

In this dialog box, follow the procedure below to change memory values collectively.

Figure 2-103. Changing Memory Contents Collectively (Memory Initialize Dialog Box)



(a) Specifying the [Start address/symbol] and [End address/symbol]

Specify the [Start address/ symbol] and [End address/ symbol] to set an address range in which you want memory contents to be initialized. Directly enter address expressions in the respective text boxes (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries). The calculation results of the address expressions you've entered are respectively handled as the start address and end address.

No address values can be specified that are greater than the address space of the microcontroller.

Caution Note that an address range that covers areas with different endians cannot be specified.

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 [Symbol name completion function](#)").

(b) Specifying the [Initialize data]

Specify the initialization data to be written into memory.

Directly enter a hexadecimal value in the text box or select an input history item from the drop-down list (up to 10 history entries).

To specify multiple pieces of initialization data, specify a maximum of 16 pieces of up to 4 bytes (8 characters) of data by separating each with a space.

Each piece of initialization data are interpreted as comprising 1 byte in units of 2 characters from the tail end of the character string. If the data consists of an odd number of characters, the first character in it is assumed to be comprising 1 byte.

Values consisting of 2 bytes or more will be converted to the arrays of bytes that match the endians in the address range to be initialized before being written to the target memory.

Input Character String (Initialize data)	Written-in image (in bytes)	
	Little endian	Big endian
1	01	01
0 12	00 12	00 12
00 012 345	00 12 00 45 03	00 00 12 03 45
000 12 000345	00 00 12 45 03 00	00 00 12 00 03 45

(c) Clicking the [OK] button

Click the [OK] button.

A pattern of the specified initialization data is written into the specified address range of memory repeatedly. (If the end address is reached in the middle of this pattern, the write process is terminated.)

However, if an invalid value or address expression is specified, CubeSuite+ displays a message and does not initialize memory values.

(7) Saving displayed memory contents

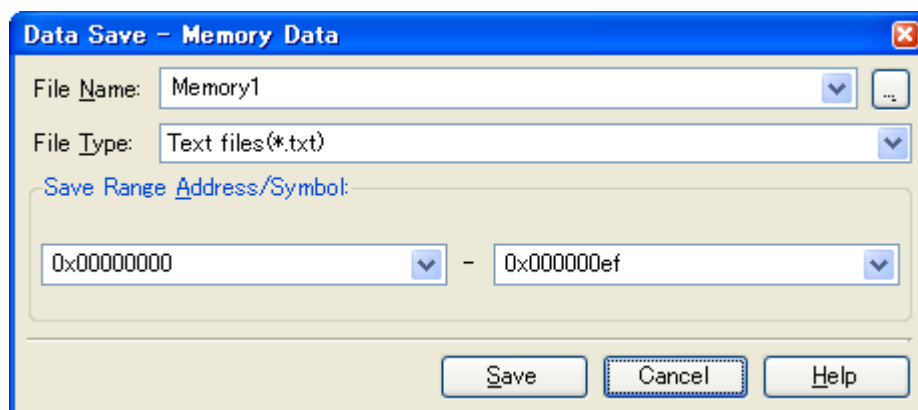
It is possible to save a specified range of memory contents to a text file (*.txt) or a CSV file (*.csv).

When saving to a file, CubeSuite+ gets latest information from the debug tool and saves data in an appropriate form in which data are currently displayed on this panel.

Choose [Save Memory Data As ...] from the [File] menu, and the [Data Save dialog box](#) shown below will be opened. (At this time, if a range is selected on panel while you choose this menu, it is possible to save only the selected range of memory data.)

In this dialog box, follow the procedure below to save memory contents.

Figure 2-104. Saving Memory Data (Data Save Dialog Box)

**(a) Specifying the [File Name]**

Specify a file name in which you want to save.

Directly enter it in the text box (specifiable in up to 259 characters) or select an input history item from the drop-down list (up to 10 history entries).

Also, you can select a file from the [Select Data Save File dialog box](#) that is opened by clicking the [...] button.

(b) Specifying the [File Type]

Select the type of file in which you want to save from the drop-down list below.

The selectable file types are as follows:

List display	Format
Text files (*.txt)	Text format (default)
CSV (comma-separated Variables) (*.csv)	CSV format Note

Note Each piece of data are separated with a comma (,) when saved.

To avoid the problem of an invalid file format in cases where any data includes a comma (,), each piece of data are enclosed in double-quotes (" ") when they are output to a file.

(c) Specifying the [Save Range Address/Symbol]

Specify the "start address" and "end addresses" to set a range of memory to be saved in a file.

Directly enter hexadecimal value or address expressions in the respective text boxes or select an input history item from the drop-down list (up to 10 history entries).

Note that if a range is selected on panel, this selected range is specified, by default, in the text boxes. If no range is selected, the currently displayed range on panel is specified.

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 Symbol name completion function").

(d) Clicking the [Save] button

Memory data is saved in specified form to a specified file.

Figure 2-105. Memory Data Output Image When Saved

[Saved to Text File (*.txt)]

(Example for data displayed in hexadecimal, 8-bit width and ASCII code)

```

+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f
00000000| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....
00000010| 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 | .....
00000020| 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 | .....
00000030| 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 | 3333333333333333

```

[Saved to CSV File (*.csv)]

(Example for data displayed in hexadecimal, 8-bit width and ASCII code)

```

00000000,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,.....
00000010,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,.....
00000020,22,22,22,22,22,22,22,22,22,22,22,22,22,22,22,.....
00000030,33,33,33,33,33,33,33,33,33,33,33,33,33,33,33,3333333333333333

```

Remark If panel contents are saved over an existing file by selecting [Save Memory Data] on [File] menu, the respective [Memory panels](#) (Memory1 - 4) are handled individually.
Also, as to the save range, the previously specified address range is applied when data is saved.

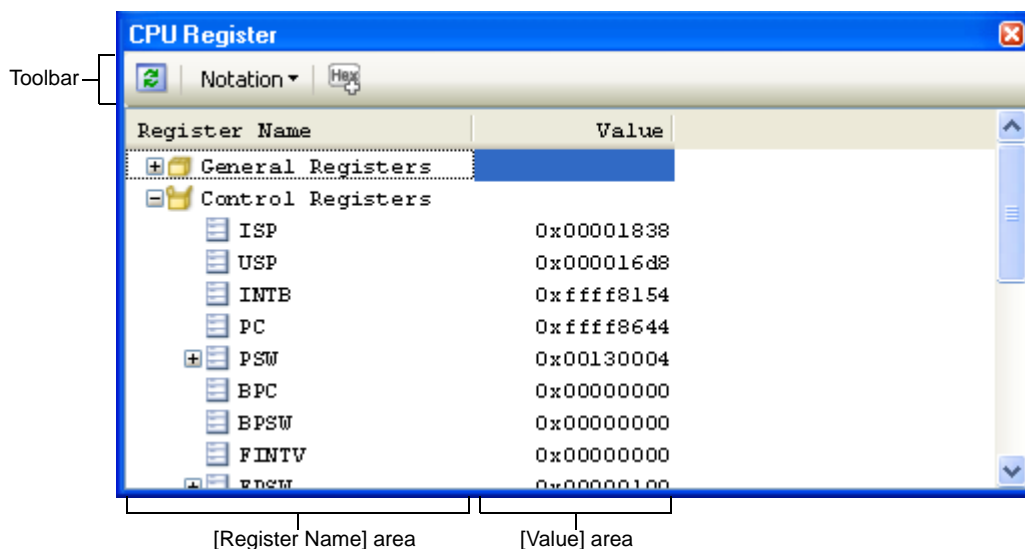
2.11.2 Displaying and changing the CPU registers

Use the [CPU Register panel](#) below to display the contents of CPU registers (general and control registers) or change the register values.

Choose [CPU Register] from the [View] menu.

For details on how to read each area and details about their functionality, see the section where the [CPU Register panel](#) is described.

Figure 2-106. Displaying the CPU Register Contents (CPU Register Panel)













Following methods of operation are described here.

- (1) [Changing the form in which values are displayed](#)
- (2) [Changing the CPU register contents](#)
- (3) [Displaying and changing the CPU register contents during program execution](#)
- (4) [Saving the displayed CPU register contents](#)

(1) Changing the form in which values are displayed

The form in which the data in the [Value] area is displayed can be freely changed using the toolbar buttons shown below.

Notation	Shows the following buttons that change the form in which values are displayed.
	Displays the value of a selected item (including low-order item) in predetermined notation (default).
	Displays the value of a selected item (including low-order item) in hexadecimal.
	Displays the value of a selected item (including low-order item) in signed decimal.
	Displays the value of a selected item (including low-order item) in unsigned decimal.
	Displays the value of a selected item (including low-order item) in octal.
	Displays the value of the selected item (including sub-items) in binary.
	Displays the character string of a selected item (including low-order item) in ASCII code. If the subject consists of 2 bytes or more, characters consisting of 1 byte each are displayed one next to another.
	Displays the value of the selected item in Float. Note that when the value is not 4-byte data, displays it in the default notation.
	Displays a selected item in Double. Except for 8-byte data, however, they are displayed in predetermined notation.
	Adds a hexadecimal equivalent for the displayed value at the end of it, with the equivalent enclosed in parentheses ().

(2) Changing the CPU register contents

The CPU register values can be edited.

Double-click the subject CPU register value in the [Value] area, and the selected value will be placed in edit mode.
(Pressing the [Esc] key cancels the edit mode.)

Edit the value directly from the keyboard and then press the [Enter] key. The value you've changed is written into the debug tool's target memory.

Caution This operation cannot be performed during program execution.

(3) Displaying and changing the CPU register contents during program execution

Select a CPU register that is the subject you want to display or change and register it as a watch expression on [Watch panel](#). That way, you can display or change the value of any CPU register in real time even while the program is running, not just when the program is halted.

For details about watch expressions, see Section ["2.11.6 Displaying and changing watch expressions"](#).

(4) Saving the displayed CPU register contents

By choosing [Save CPU Register Data As...] from the [File] menu, it is possible to open the [Save As dialog box](#) and then save the contents of CPU registers in whole to a text file (*.txt) or a CSV file (*.csv).

When saving to a file, CubeSuite+ gets latest information from the debug tool.

Figure 2-107. CPU Register Value Output Image When Saved

Register name	Value

Category name	
-Register name	Value
:	:

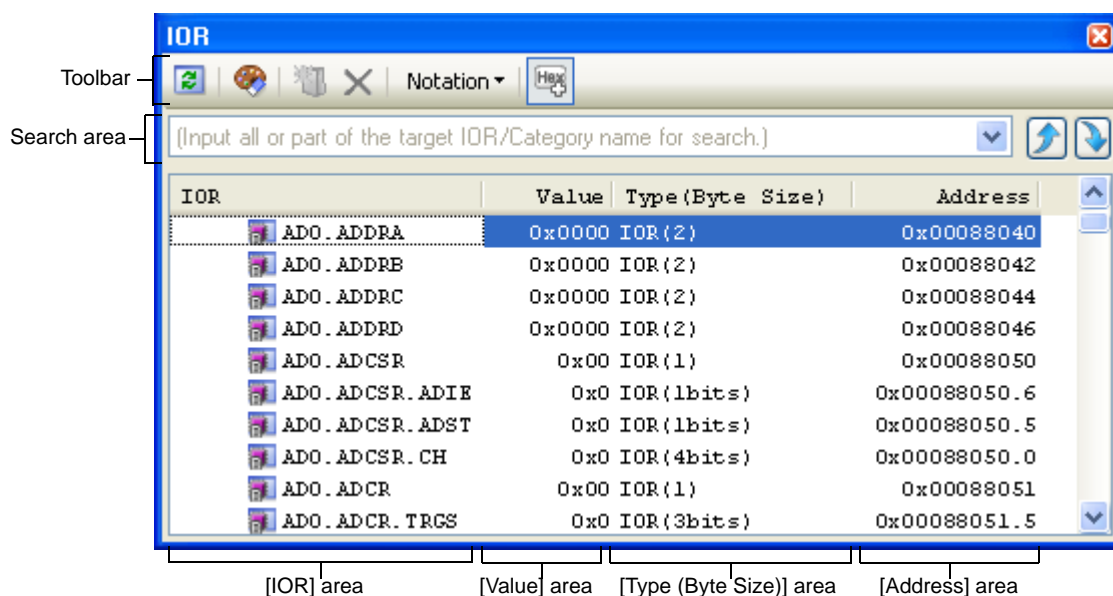
2.11.3 Displaying and changing the I/O registers

Use the [IOR panel](#) shown below to display I/O register contents or change values.

Choose [IOR] from the [View] menu.

For details on how to read each area and details about their functionality, see the section where the [IOR panel](#) is described.

Figure 2-108. Displaying the I/O Register Contents (IOR Panel)



Following methods of operation are described here.

- (1) Searching I/O register
- (2) Putting the I/O registers in order
- (3) Changing the form in which values are displayed
- (4) Changing the contents of I/O registers
- (5) Displaying and changing I/O register contents during program execution
- (6) Saving the displayed I/O register contents



(1) Searching I/O register



You can search the name of I/O register.

In the search area, specify the name of I/O register (case-insensitive) in the text box.

You can either enter characters directly from the keyboard (up to 512 characters) or select from the input history items in the drop-down list (up to 10 items).

Click one of the following buttons.

	Searches I/O register names containing the character string specified in the text box in the backward direction and highlights the search result.
	Searches I/O register names containing the character string specified in the text box in the forward direction and highlights the search result.


- Remarks 1.** I/O register names which are categorized in folders and hidden are also searched (They will be expanded and selected.).
- 2.** After entering the search string, pressing [Enter] key will perform the same function as clicking on the  button, and pressing [Shift]+[Enter] will perform the same function as clicking on the  button.

(2) Putting the I/O registers in order

The tree form of I/O registers can be edited by classifying each register by a given category (folder).

- Cautions 1.** No other categories can be created within a category.
- 2.** No I/O registers can be added or removed.


(a) To create a new category

Move the caret to an I/O register name you want to create and then click the toolbar button  and enter a new category name directly from the keyboard.

(b) To edit a category name

Select a category name you want to edit and then click on it again and edit the category name directly from the keyboard.

(c) To remove a category








Select a category you want to remove and then click the toolbar button  However, only blank categories can be removed.

(d) To change the order in which I/O registers are displayed

Drag-and-drop an I/O register name into any category. That way, the I/O registers are classified by a category. Also, the order in which categories and I/O register names are displayed (one above or below another) can be freely changed by a drag-and-drop operation.

(3) Changing the form in which values are displayed

The form in which the data in the [Value] area is displayed can be freely changed using the toolbar buttons shown below.

Notation	Shows the following buttons that change the form in which values are displayed.
	Displays the value of a selected item in hexadecimal (default).
	Displays the value of a selected item in signed decimal.
	Displays the value of a selected item in unsigned decimal.
	Displays the value of a selected item in octal.
	Displays the value of a selected item in binary.
	Displays the value of a selected item in ASCII code.
	Adds a hexadecimal equivalent for the displayed value of a selected item at the end of the value, with the equivalent enclosed in parentheses ().

(4) Changing the contents of I/O registers

The I/O register values can be edited.

In the [Value] area, click the selected I/O register value again to set it in the Edit mode (Pressing the [Esc] key cancels the edit mode).

Edit the value directly from the keyboard and then press the [Enter] key. The value you've changed is written into the debug tool's target memory.

- Cautions 1.** This operation cannot be performed during program execution.
- 2.** The values of read-only I/O registers cannot be changed.

- Remarks 1.** If a number with fewer digits than the size of the I/O register is entered, the higher-order digits will be padded with zeroes.
- 2.** If a number with more digits than the size of the I/O register is entered, the higher-order digits will be masked.
- 3.** I/O register values can also be entered using ASCII characters.
- When "0x41" is entered as the value of I/O register "CRC.CRCDOR"
 - >> "0x41" is written to "CRC.CRCDOR".
 - When ASCII character "A" is entered as the value of I/O register "CRC.CRCDOR"
 - >> "0x41" is written to "CRC.CRCDOR".

(5) Displaying and changing I/O register contents during program execution

Select an I/O register that is the subject you want to display or change and register it as a watch expression on [Watch panel](#). That way, you can display or change the value of any I/O register in real time even while the program is running, not just when the program is halted.

For details about watch expressions, see Section ["2.11.6 Displaying and changing watch expressions"](#).

(6) Saving the displayed I/O register contents

By choosing [Save IOR Data As...] from the [File] menu, it is possible to open the [Save As dialog box](#) and then save the contents of I/O registers in whole to a text file (*.txt) or a CSV file (*.csv) (Regardless of whether you've chosen to show or not show on this panel, the values of all I/O registers are saved.).

When saving to a file, CubeSuite+ reloads I/O register values and saves the latest values thus obtained.

However, the I/O registers protected against read are not reloaded. To save the latest content, select [Force Read Value] from the context menu before saving to a file.

Figure 2-109. I/O Register Value Output Image When Saved

IOR name	Value	Type (Byte Size)	Address

Category name			
-IOR name	Value	Type (Byte Size)	Address
:	:	:	:

2.11.4 Displaying and changing global and static variables

Use the [Watch panel](#) to display or change global variables or static variables.

Select a variable whose value you want to display or change and register it as a watch expression on watch panel.

For details about watch expressions, see Section "2.11.6 [Displaying and changing watch expressions](#)".

2.11.5 Displaying and changing local variables

Use the [Local Variables panel](#) shown below to display the contents of local variables and change values.

Choose [Local Variables] from the [View] menu.

To display the content of your desired local variable, select a scope in the scope area.

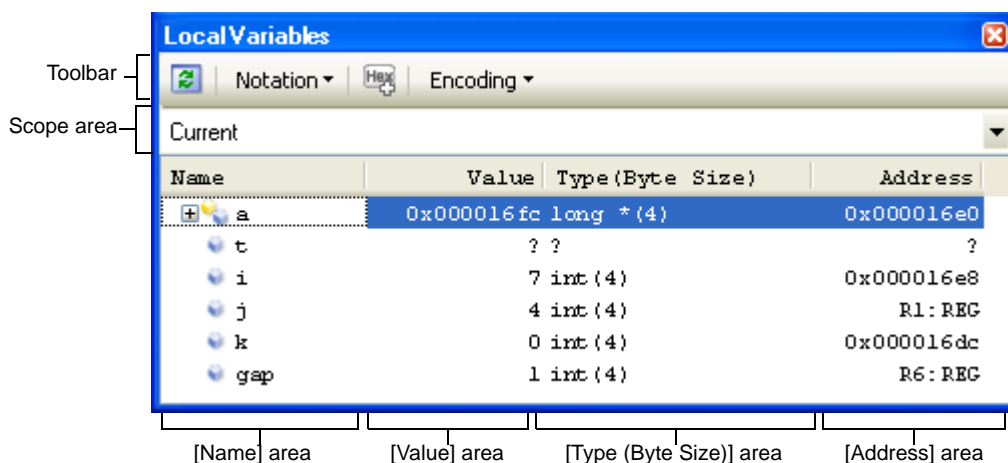
The local variable panel shows local variable names and function names. It also shows parameters to functions as local variables.

For details on how to read each area and details about their functionality, see the section where the [Local Variables panel](#) is described.

Caution During program execution, nothing is displayed on this panel.

Each area on this panel are displayed at the time the program has stopped running.

Figure 2-110. Displaying the Contents of Local Variables (Local Variables Panel)


















Following methods of operation are described here.

- (1) [Changing the form in which values are displayed](#)
- (2) [Changing the contents of local variables](#)
- (3) [Saving the displayed contents of local variables](#)

(1) Changing the form in which values are displayed

The form in which the data in the [Value] area are displayed can be freely changed using the toolbar buttons shown below.

Notation	Shows the following buttons that change the form in which values are displayed.
	Displays values on this panel in per-variable predetermined notation (default).
	Displays values on this panel in hexadecimal.
	Displays values on this panel in decimal.
	Displays values on this panel in octal.
	Displays values on this panel in binary.
	Displays array indexes on this panel in decimal (default).
	Displays array indexes on this panel in hexadecimal.
	Displays values on this panel in Float. Except for 4-byte data or when values have type information, however, they are displayed in predetermined notation.
	Displays values on this panel in Double. Except for 8-byte data or when values have type information, however, they are displayed in predetermined notation.
	Adds a hexadecimal equivalent for the displayed value at the end of it, with the equivalent enclosed in parentheses ().
Encoding	Shows the following buttons that change the encode in which string variables are displayed.
	Displays string variables in ASCII code (default).
	Displays string variables in Shift_JIS code.
	Displays string variables in EUC-JP code.
	Displays string variables in UTF-8 code.
	Displays string variables in UTF-16 code.

(2) Changing the contents of local variables

The values of local variables and parameter values can be edited.

Select the value of the subject local variable or the value of parameters to it in the [Value] area and click on it again.

The selected value is placed in edit mode. (Pressing the [Esc] key cancels the edit mode.)

Enter a value directly from the keyboard and then hit the [Enter] key. The value you've changed is written into the debug tool's target memory. At this time, CubeSuite+ checks the value to see if it fits to type. If inappropriate, the editing you've done is ignored.

Caution This operation cannot be performed during program execution.

- Remarks 1.** If the numeral entered for a variable is smaller than the size of the variable, its high-order digits are padded with zeroes.
- 2.** If the numeral entered for a variable is larger than the size of the variable, its high-order digits are masked.
- 3.** For character arrays (char type or unsigned char type), if ASCII is selected for the display form, it is possible to use character strings (ASCII, Shift_JIS, EUC-JP, or Unicode (UTF-8/UTF-16)) to enter values.

4. Local variables can also be entered using ASCII characters.
 - When using ASCII characters to enter values
Enter the letter "A" in the [Value] area for the variable "ch".
>>The numeral "0x41" is written into the memory area to which the variable "ch" is mapped.
 - When using numerals to enter values
Enter the numeral "0x41" in the [Value] area for the variable "ch".
>> The numeral "0x41" is written into the memory area to which the variable "ch" is mapped.
 - When using character strings (ASCII) to enter values
Set the display form of the character array "str" to ASCII and then enter the letters "ABC" in the [Value] area.
>> The numerals "0x41," "0x42," "0x43" and "0x00" are written into the memory area to which the array "str" is mapped.

(3) Saving the displayed contents of local variables

By choosing [Save Local Variables Data As...] from the [File] menu, it is possible to open the [Save As dialog box](#) and then save the contents of local variables in whole to a text file (*.txt) or a CSV file (*.csv).

When saving to a file, CubeSuite+ gets latest information from the debug tool.

Note that if arrays, pointer-type variables, structures/unions, or CPU registers (only those that are assigned the names to represent sections) are displayed in expanded form, the values of their expansion elements are also saved. When not expanded, they are marked with "+" at the head, with the values left blank.

Figure 2-111. Local Variable Value Output Image When Saved

Scope : <i>Current scope</i>				
[V]Variable	[P]Parameter	[F]Function		
Name	Value	Type	(Byte Size)	Address

[V]Variable name[1]	Value	Type		Address
- [V]Variable name[0]	Value	Type		Address
:	:	:		:

2.11.6 Displaying and changing watch expressions

C variables, CPU registers, I/O registers, and assembler symbols can be registered as watch expressions in the [Watch panel](#) shown below. That way, it is possible to get their values from the debug tool at all times and, thereby, to keep watch on values.

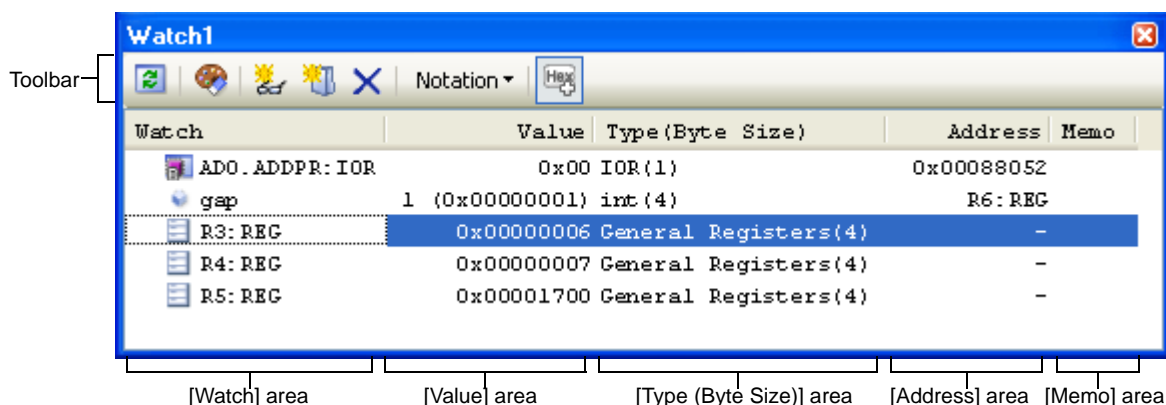
Also, the watch expressions permit display of values to be updated successively, even while the program is under execution (see section "(7) [Displaying and changing the contents of watch expressions during program execution](#)").

The watch panel is opened by choosing [Watch] from the [View] menu and then selecting [Watch 1-4].

Up to four pieces of watch panels can be opened at a time. Each panel is discriminated by the name "Watch 1," "Watch 2," "Watch 3," and "Watch 4" in the title bar. The respective watch panels have their watch expressions registered and managed individually and saved as user information for the project.

For details on how to read each area and details about their functionality, see the section where the [Watch panel](#) is described.

Figure 2-112. Displaying the Contents of Watch Expression (Watch panel)



Following methods of operation are described here.

- (1) Registering watch expressions
- (2) Putting the registered watch expressions in order
- (3) Editing the registered watch expressions
- (4) Removing watch expressions
- (5) Changing the form in which values are displayed
- (6) Changing the contents of watch expressions
- (7) Displaying and changing the contents of watch expressions during program execution
- (8) Saving the displayed contents of watch expressions

(1) Registering watch expressions

There are following three methods to register watch expressions (By default, no watch expressions are registered.).

- Cautions 1.** Up to 128 watch expressions can be registered in one **Watch panel** (If an attempt is made to register beyond the upper limit, a message is displayed).
- 2.** Because of optimization by a compiler, for blocks where variables, or the subject to be operated on, are not in use, there may be no variable data in the stack and registers. In this case, those unused variables, even when registered as watch expressions, will have their displayed values marked with "?".

- Remarks 1.** The respective watch expressions registered in each watch panel (watch 1 through watch 4) are managed individually and saved as user information for the project.
- 2.** Plural watch expressions with the same name can be registered.

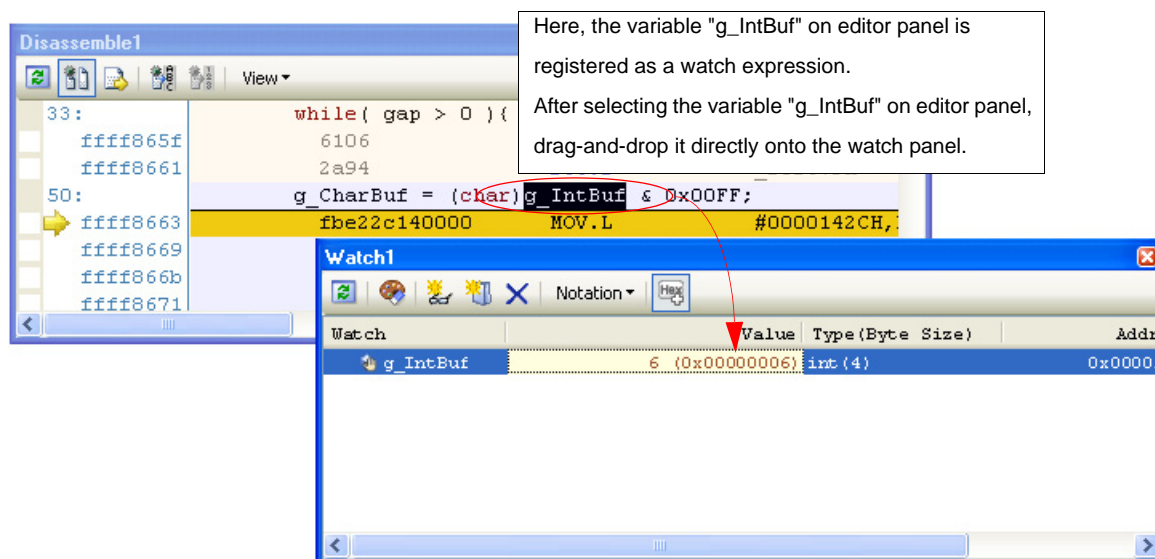
(a) To register from other panels

Watch expressions can be registered from other panels of CubeSuite+.

In one of other panels, select the subject you want to register as a watch expression and drag-and-drop it directly onto any watch panel (watch 1 through watch 4).

Note that there is certain relationship between the panels that accept this operation and the subjects that can be registered as watch expressions. For details, see "[Table A-4. Relationship between Panels and Targets That Can be Registered as Watch Expressions](#)".

Figure 2-113. Example for Registering Watch Expressions from Other Panels



Remark There is an alternative way to register watch expressions. Select the subject you want to register as a watch expression, or move the caret to one of the subject character strings (the subject being automatically determined), then select [Register In Watch 1] from the context menu (However, this method is usable for only the [Watch panel](#) (Watch 1)).

(b) To register directly on watch panel


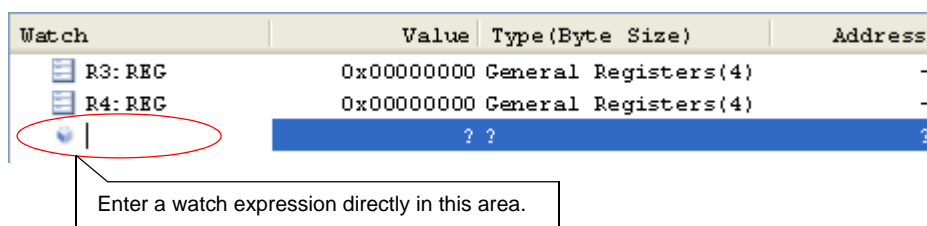
In any [Watch panel](#) (watch 1 through watch 4), click the toolbar button  and the entry box shown below will be displayed in the [Watch] area.

Figure 2-114. Entry Box for Watch Expressions



Enter a watch expression in the entry box directly from the keyboard and then hit the [Enter] key.
For the input forms of watch expressions entered this way, see the tables listed below.

- ["Table A-5. Input Format of Watch Expression"](#)
- ["Table A-7. Scope Specification of C/C++ Language Variable Used with Watch Expression Registration"](#)
- ["Table A-8. Scope Specification of CPU Register with Watch Expression Registration"](#)
- ["Table A-9. Scope Specification of I/O register with Watch-Expression Registration"](#)

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see ["2.20.2 Symbol name completion function"](#)).

(c) To register from other applications

Select a C variable, CPU register, I/O register, or assembler symbol character string from an external editor or the like and then drag-and-drop it directly onto the [Watch panel](#) (Watch 1 through Watch 4).

In this case, the character string you've dropped is registered as a watch expression directly as it is.


(2) Putting the registered watch expressions in order

The registered watch expressions can be classified by a category (folder) for display in tree form (By default, there are no categories.).

Cautions 1. No other categories can be created within a category.

2. Up to 64 categories can be created in one watch panel. (If an attempt is made to create beyond the upper limit, a message is displayed.).


(a) To create a new category

Move the caret to the position at which you want to create and then click the toolbar button  and enter a new category name directly from the keyboard.

(b) To edit a category name

Select a category name you want to edit and then click on it again and edit the category name directly from the keyboard.

(c) To remove a category

Select a category you want to remove and then click the toolbar button  .

(d) To change the order in which watch expressions are displayed

Drag-and-drop a registered watch expression directly onto the category you've created. That way, watch expressions are classified by a category.

Also, the order in which categories and watch expressions are displayed (one above or below another) can be freely changed by a drag-and-drop operation.

Remark When you drag-and-drop a watch expression or category onto the watch panel (watch 1 through watch 4), the watch expression or category is copied to the watch panel onto which you've dropped.


(3) Editing the registered watch expressions

The watch expressions you've registered can be edited.

Double-click on a watch expression you want to edit, and the subject watch expression will be placed in edit mode. (Pressing the [Esc] key cancels the edit mode.)











Edit the content directly from the keyboard and hit the [Enter] key.

(4) Removing watch expressions

To remove a registered watch expression, select the watch expression on [Watch panel](#) that you want to remove and then click the toolbar button  .

(5) Changing the form in which values are displayed

The form in which the data in the [Value] area is displayed can be freely changed using the toolbar buttons shown below.

Notation	Shows the following buttons that change the form in which values are displayed.
	Displays the value of a selected watch expression in per-variable predetermined notation (default) (see "Table A-10. Display Format of Watch Expressions (Default)").
	Displays the value of a selected item in hexadecimal.
	Displays the value of a selected item in signed decimal.
	Displays the value of a selected item in unsigned decimal.
	Displays the value of a selected item in octal.
	Displays the value of a selected item in binary.
	Displays the value of a selected item in ASCII code.
	Displays a selected item in Float. However, this is valid only when the selected watch expression consists of 4-byte data.
	Displays a selected item in Double. However, this is valid only when the selected watch expression consists of 8-byte data.
	Adds a hexadecimal equivalent for the displayed value of a selected item at the end of the value, with the equivalent enclosed in parentheses (). However, this information is not added when values are displayed in hexadecimal.

(6) Changing the contents of watch expressions

The values of watch expressions can be edited.

Double-click the value of the watch expression in the [Value] area that you want to edit, and the value you've clicked is placed in edit mode. (Pressing the [Esc] key cancels the edit mode.)

Edit the value directly from the keyboard and then hit the [Enter] key. The value you've changed is written into the debug tool's target memory.

However, only the watch expressions that correspond one for one to C variables, CPU registers, I/O registers, or assembler symbols can have their values changed. Nor can the values of read-only I/O registers be changed.

This operation can be taken place while the program is in execution. See "(4) Displaying and changing memory contents during program execution" for details on how to operate it.

- Remarks 1.** If the numeral entered for a variable is smaller than the size of the variable, its high-order digits are padded with zeroes.
- If the numeral entered for a variable is larger than the size of the variable, its high-order digits are masked.
 - For character arrays (char type or unsigned char type), if ASCII is selected for the display form, it is possible to use character strings (ASCII, Shift_JIS, EUC-JP, or Unicode (UTF-8/UTF-16)) to enter values.
 - Watch-expressions can also be entered using ASCII characters.
 - When using ASCII characters to enter values
Enter the letter "A" in the [Value] area of the variable "ch".
>> The numeral "0x41" is written into the memory area to which the variable "ch" is mapped.
 - When using numerals to enter values
Enter the numeral "0x41" in the [Value] area of the variable "ch".
>> The numeral "0x41" is written into the memory area to which the variable "ch" is mapped.
 - When using character strings (ASCII) to enter values

Set the display form of the character array "str" to ASCII and enter the letters "ABC" in the [Value] area.

>> The numerals "0x41," "0x42," "0x43" and "0x00" are written into the memory area to which the array "str" is mapped.

(7) Displaying and changing the contents of watch expressions during program execution

The [Memory panel](#) and [Watch panel](#) come with a realtime display update function that permits you to update display of, or even rewrite, the contents of memory or watch expressions in real time.

By enabling this realtime display update function, it is possible to display or change the values of memory or watch expressions even while the program is running, not just when the program is halted.

For details on how to set, see section "(4) [Displaying and changing memory contents during program execution](#)"

(8) Saving the displayed contents of watch expressions

By choosing [Save Watch Data As...] from the [File] menu, it is possible to open the [Save As dialog box](#) and then save the contents of watch expressions and values in whole to a text file (*.txt) or a CSV file (*.csv).

When saving to a file, CubeSuite+ reloads the values of watch expressions and saves the latest values thus obtained.

Note that if arrays, pointer-type variables, structures/unions, or CPU registers (only those that are assigned section names) are displayed in expanded form, the values of their expansion elements are also saved. When not expanded, they are marked with "+" at the head, with the values left blank.

However, the I/O registers protected against read are not reloaded. To save the latest content, select [Force Read Value] from the context menu before saving to a file.

Figure 2-115. Watch Data Output Image When Saved

Watch-expression	Value	Type(Byte Size)	Address	Memo
Watch-expression	Value	Type(Byte Size)	Address	Memo
-Category name				
Watch-expression	Value	Type(Byte Size)	Address	Memo
:	:	:	:	:

Remark If panel contents are saved over an existing file by selecting [Save Watch Data] on [File] menu, the respective watch panels (watch 1-4) are handled individually.

2.12 Display Function Call Information from the Stack

This section describes how to display function-call information from the stack.

The compiler bundled with CubeSuite+ (CC-RX) places function-call information on the stack in line with the ANSI standards. By analyzing this function-call information (hereafter referred to as call-stack information), it is possible to know the depth of function calls, the positions from which calls are made, and parameters to those functions.

2.12.1 Display call stack information

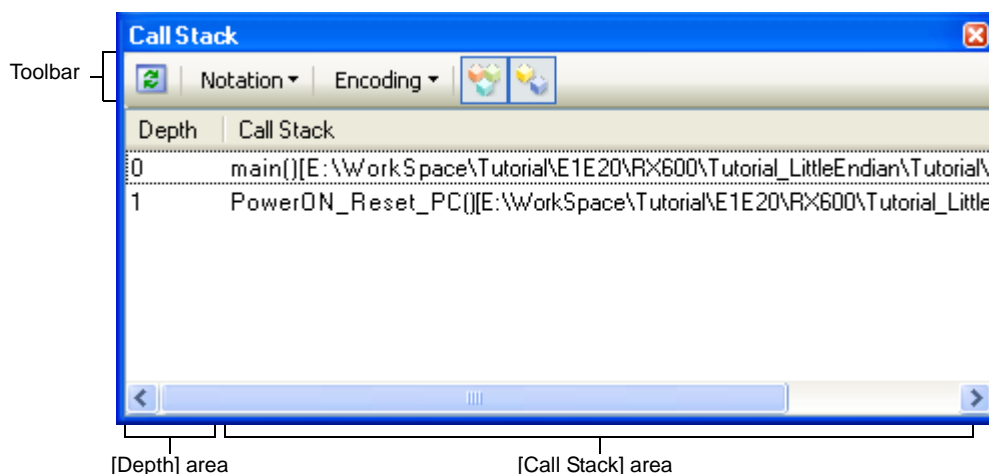
Use the [Call Stack panel](#) shown below to display call-stack information.

Select [Call Stack] from the [View] menu.

For details on the contents and function in each area, see the section for the [Call Stack panel](#).

Caution This panel is left blank while the program is in execution.
Each area on this panel are displayed at the time the program has stopped running.

Figure 2-116. Display Call Stack Information (Call Stack Panel)













This section describes the following.

- (1) [Changing the form in which values are displayed](#)
- (2) [Jumping to the source line](#)
- (3) [Displaying local variables](#)
- (4) [Saving the displayed contents of call-stack information](#)

(1) Changing the form in which values are displayed

The form in which values on this panel are displayed can be freely changed using the toolbar buttons shown below. However, this feature is disabled during program execution.

Notation	Shows the following buttons that change the form in which values are displayed.
	Displays values on this panel in per-variable predetermined notation (default).
	Displays values on this panel in hexadecimal.
	Displays values on this panel in decimal.
	Displays values on this panel in octal.
	Displays values on this panel in binary.
Encoding	Shows the following buttons that change the encode in which string variables are displayed.
	Displays string variables on this panel in ASCII code (default).
	Displays string variables on this panel in Shift_JIS code.
	Displays string variables on this panel in EUC-JP code.
	Displays string variables on this panel in UTF-8 code.
	Displays string variables on this panel in UTF-16 code.

(2) Jumping to the source line

Double-clicking on a line will open the [Editor panel](#), with the caret moved to the function call source line indicated by the selected line. (The view will jump to the Editor panel if it is already open.)

Remark Selecting [Jump to Disassemble] from the context menu will open the [Disassemble panel](#) (Disassemble1), with the caret moved to the function call source address indicated by the selected line. (The view jumps to the Disassemble panel (Disassemble1) if it is already open.)

(3) Displaying local variables

Selecting [Jump to Local Variable at This Time] from the context menu will open the [Local Variables panel](#) which displays local variables of the function indicated by the selected line.

(4) Saving the displayed contents of call-stack information

Selecting [Save Call Stack Data As...] from the [File] menu will open the [Save As dialog box](#) box in which you can save the entire call stack information either in a text file (*.txt) or a CSV file (*.CSV) format. When saving to a file, the latest information will be retrieved from the debug tool.

Figure 2-117. Call Stack Information Output Image When Saved

Depth	Call stack
0	Call stack information
1	Call stack information
:	:

2.13 Collecting an Execution History

This section describes how to collect an execution history of the program.

Generally, a program's execution history is referred to as a "trace", the term of which is used in the pages below. If the program goes wild, it is very difficult to find the cause of the problem from the memory contents or stack information after the program has run out of control. However, analysis of the content of collected trace data makes it possible to explore a process of malfunction until the program starts running wild, providing an effective means for discovering potential bugs in the program.

Caution [E20(JTAG) [RX600 Series]]

Part of the trace functions and Real-time RAM Monitor (RRM) functions can be used only on a mutually exclusive basis.

2.13.1 Setting up a trace operation

When the trace function begins, trace data which has recorded in it an execution history of the currently executed program is collected in trace memory (when program execution stops, the trace function also automatically stops).

Before the trace function can be used, it is necessary to make settings relating to the operation of a trace.

Note that the method on how to set differs with each debug tool used.

- (1) For [E1]
- (2) For [E20]
- (3) For [Simulator]

(1) For [E1]

Make settings in the [\[Trace\]](#) category on the [Property panel's \[Debug Tool Settings\] tab](#).

Figure 2-118. [Trace] Category [E1]

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch

(a) [Usage of trace function]

Only the trace function can be used. Do not specify [Real-time RAM monitor] for this property.

(b) [Operation after trace memory is full]

Specify the operation to be performed when the trace memory is filled with collected trace data, from the drop-down list below.

Overwrite trace memory and continue execution	When the trace memory is filled, CubeSuite+ continues writing trace data over the old data (default).
Stop trace	When the trace memory is filled, CubeSuite+ stops writing trace data (but does not stop program execution).
Stop	When the trace memory is filled, CubeSuite+ stops writing trace data and stops program execution at the same time.

(c) [Trace data type]

For this property, specify the type of trace data to be collected from the drop-down list below.

Branch	Source/destination address information of branching during program execution are collected as trace data.
Branch + Data access ^{Note 1}	Source/destination address information of branching during program execution, as well as data information on access events that occurred are collected as trace data.
Data access ^{Note 2}	Data information on access events that occurred during program execution are collected as trace data.

Notes 1. [E1(Serial) [RX200 Series]]

Trace data for [Branch + Data access] cannot be collected. Therefore, this item is not displayed in the drop-down list.

2. [E1(Serial) [RX200 Series]]

To collect trace data for [Data access], it is necessary to set address conditions in a point trace. For details about the point trace, see "2.13.4 Collecting an execution history only when conditions are met".

(2) For [E20]

Make settings in the [Trace] category on the Property panel's [Debug Tool Settings] tab.

Figure 2-119. [Trace] Category [E20]

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch
External trace output	CPU execution
Trace memory size[MByte]	1

(a) [Usage of trace function]

Part of the trace functions and Real-time RAM Monitor (RRM) functions can be used only on a mutually exclusive basis. Therefore, in this property, specify which function you want to be used preferentially. Here, select [Trace] from the drop-down list below.

Trace	Uses the trace function preferentially (default). - Real-time RAM monitor function cannot be used.
Real-time RAM monitor ^{Note 1}	Uses the real-time RAM monitor function (RRM) preferentially. - Trace function Use subject to limitations ^{Note 2} Also, trace-related events are disabled.

Notes 1. [E20(Serial)]

The real-time RAM monitor function is not supported. Therefore, do not specify [Real-time RAM monitor] for this property value.

2. [E20(JTAG) [RX600 Series]]

Part of the trace function cannot be used.
Following limitations apply.

Operation after trace memory is full	[Stop trace] and [Stop] cannot be used. Only [Overwrite trace memory and continue execution] can be used.
Trace data type	[Branch] and [Branch + Data access] cannot be used. Only [Data access] can be used.
External trace output	[Do not output] cannot be used. [CPU execution] and [Trace output] can be used.
Trace memory size[MByte]	Only 1M byte can be used.

(b) [Operation after trace memory is full]

Specify the operation to be performed when the trace memory is filled with collected trace data, from the drop-down list below.

Overwrite trace memory and continue execution	When the trace memory is filled, CubeSuite+ continues writing trace data over the old data (default).
Stop trace	When the trace memory is filled, CubeSuite+ stops writing trace data (but does not stop program execution).
Stop	When the trace memory is filled, CubeSuite+ stops writing trace data and stops program execution at the same time.

(c) [Trace data type]

This property is displayed only when you've selected [Trace] for the [\[Usage of trace function\]](#) property. Specify the type of trace data to be collected, from the drop-down list below.

Branch	Source/destination address information of branching during program execution are collected as trace data.
Branch + Data access ^{Note 1}	Source/destination address information of branching during program execution, as well as data information on access events that occurred are collected as trace data.
Data access ^{Note 2}	Data information on access events that occurred during program execution are collected as trace data.

Notes 1. [E20(Serial) [RX200 Series]]

Trace data for [Branch + Data access] cannot be collected. Therefore, this item is not displayed in the drop-down list.

2. [E20(Serial) [RX200 Series]]

To collect trace data for [Data access], it is necessary to set address conditions in a point trace. For details about the point trace, see [“2.13.4 Collecting an execution history only when conditions are met”](#).

(d) [External trace output] [E20(JTAG)]

Specify the method on how the collected trace data should be output from the drop-down list below.

CPU execution	CPU execution given priority over trace output. Trace information may be lost if output.
Trace output	Trace output given priority over CPU execution. CPU execution stops during trace output, affecting real-time performance.
Do not output	Only the internal buffer of the microcomputer will be used, with no output of trace information.

(e) [Trace memory size[MByte]] [E20(JTAG)]

Specify the size of memory used to retain trace data from the drop-down list below.

- 1 (default), 2, 4, 8, 16, 32

(3) For [Simulator]

Make settings in the [\[Trace\]](#) category on the [Property panel](#)'s [\[Debug Tool Settings\]](#) tab.

Figure 2-120. [Trace] Category [Simulator]

Trace	
Use trace function	No
Clear trace memory before running	Yes
Operation after trace memory is full	Non stop and overwrite to trace memory
Accumulate trace time	No
Trace memory size[frames]	64K

(a) [Use trace function]


Specify from the drop-down list whether or not to use the trace function.

To use the trace function, specify [Yes] (by default, [No] is selected).

(b) [Clear trace memory before running]

Specify from the drop-down list whether or not to clear (initialize) the trace memory once before the trace function begins.

To clear, specify [Yes] (default).

Remark The trace memory can be forcibly cleared by clicking the  button in the [Trace panel](#) toolbar.

(c) [Operation after trace memory is full]

Specify the operation to be performed when the trace memory is filled with collected trace data, from the drop-down list below.

Non stop and overwrite to trace memory	When the trace memory is filled, CubeSuite+ continues writing trace data over the old data (default). If you've selected [Yes] for the [Clear trace memory before running] property, when the program is re-executed, the trace memory is cleared before trace data is written to.
Stop	When the trace memory is filled, CubeSuite+ stops writing trace data and stops program execution at the same time. If you've selected [No] for the [Clear trace memory before running] property, even when re-executed, the program is halted and not executed.

(d) [Accumulate trace time]

Specify from the drop-down list whether or not to use an accumulated display for trace time display.

Specify [Yes] to use an accumulated display for trace time display, or [No] to use a difference display (default).

(e) [Trace memory size[frames]]

Specify from the drop-down list the size of trace memory (number of trace frames).

Note that a trace frame is a unit for the trace data, and that fetch, write and read operations each use one trace frame.

The drop-down list shows the number of trace frames as follows:

64K (default), 128K, 256K, 512K, 1M

2.13.2 Collecting an execution history up to a halt

The debug tool has a preinstalled function to collect an execution history of the program from when it starts running to when it stops.

Thanks to this, when the program starts running, collection of trace data begins automatically, and it is finished at the same time the program stops.

For details on how to check the collected trace data, see "[2.13.5 Displaying an execution history](#)".

Caution [Simulator]

For data access by string-manipulating and multiply-and-accumulate instructions, only a history of the last access is collected.

Remark This function is actuated by an unconditional trace event, one of the built-in events that are set in the debug tool by default.

Note that this unconditional trace event and the trace event described later (see "[2.13.3 Collecting an execution history in a section](#)" and "[2.13.4 Collecting an execution history only when conditions are met](#)") are used exclusively of each other. Therefore, if the trace event is **Enabled**, the unconditional trace event is automatically **Disabled**.

When no trace event has been set, trace data is collected even if the checkbox for the unconditional trace event is deselected (i.e. **Disabled**).

2.13.3 Collecting an execution history in a section

By setting a trace event, it is possible to collect only a history of execution in a given section as trace data during program execution process.

This trace event consists of a trace start event and a trace end event.

To use this function, follow the procedure described below.

- (1) [Setting a trace start event and a trace end event](#)
- (2) [Combining multiple events \[E1\] \[E20\]](#)
- (3) [Executing the program](#)
- (4) [Editing trace start and trace end events](#)
- (5) [Deleting a trace start or trace end event](#)

Cautions 1. Regarding trace event settings (e.g., limitations on the number of valid events), also see "[2.17.7 Points to note regarding event setting](#)".

2. The events that can be set as trace start and end events differ with each debug tool used.

- [E1] [E20]

execution-related and access-related events

- [Simulator]
execution-related events only

3. [Simulator]

For data access by string-manipulating and multiply-and-accumulate instructions, only a history of the last access is collected.

(1) Setting a trace start event and a trace end event

On the [Editor panel](#) or [Disassemble panel](#), set the events at which you want the collection of trace data to start and finish.

(a) For execution-related events

By setting execution-related events for the trace start and trace end events, it is possible to start and finish the collection of trace data at any place.

To set an execution-related event as the trace start event, move the caret to the line or address^{Note} at which you want the collection of trace data to start and then choose [Start Tracing] from [Trace Settings] on the context menu.

A trace start event is set for the instruction at the beginning address corresponding to the line or address at the caret position.

Also, when you set an execution-related event as the trace end event, move the caret to the line or address^{Note} at which you want the collection of trace data to finish and then choose [Stop Tracing] from [Trace Settings] on the context menu.

A trace end event is set for the instruction at the beginning address corresponding to the line or address at the caret position.

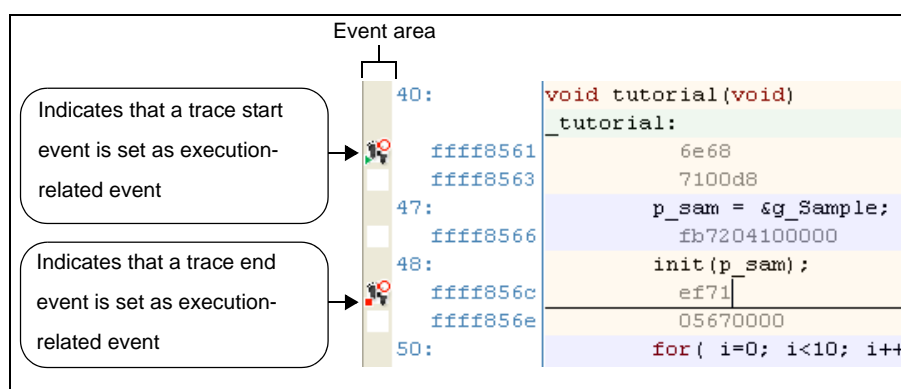
Note Trace start and trace end events cannot be set on lines that have no addresses displayed.

When trace start and trace end events are set, the following event marks are displayed in the event area of the relevant line or address.

Table 2-12. Trace Start Event and Trace End Event Marks (Execution-related Events)

Type	Event mark
Trace start	
Trace end	

Figure 2-121. Example of Trace Start and Trace End Events Set by Using Execution-related Events (For the Disassemble panel)



(b) For access-related events

By setting access-related events for the trace start and trace end events, it is possible to start and finish the collection of trace data when a specified access is made to any variable or I/O register.

At this time, it is also possible to limit the values accessed.

The types of access specifiable in access-related events are as follows:

Table 2-13. Types of Access for Variables

Access Type	Description
Read	Collection of trace data is started or finished when a specified variable or I/O register is accessed for read (i.e., to read data from it).
Write	Collection of trace data is started or finished when a specified variable or I/O register is accessed for write (i.e., to write data to it).
Read/write	Collection of trace data is started or finished when a specified variable or I/O register is accessed for read or write (i.e., to read or write data).

Remark Accesses by DMAC (Direct Memory Access Controller) and DTC (Data Transfer Controller) are not supported.

- To set events for variables or I/O registers in source text or disassembled text

Perform this operation in the [Editor panel/Disassemble panel](#) in which the source text/disassembly text is displayed.

Select any variable or I/O register in the source text or disassembled text and then select the desired operation from the context menu as described below. However, the variables you can select are only global variables, static variables in function, or static variables in file.

Note that when you have performed this operation, it is interrupted as if a trace event (access-related) has been set for the selected variable or I/O register, and it is managed on the [Events panel](#) (see “[2.17 Event Management](#)”).

Access type	Operation
Read/write	<p>[Trace start event] Choose [Record Start R/W Value] from [Trace Settings] and then hit the [Enter] key.</p> <p>[Trace end event] Choose [Record End R/W Value] from [Trace Settings] and then hit the [Enter] key.</p> <p>If a value is specified in the text box of the menu at this time, collection of trace data is started or finished only when a read/write is performed with a specified value. If no values are specified, the program breaks when a selected variable is read or written to, no matter what value is read or written.</p>

Cautions 1. Only the variables in current scope are selectable.

- 2. No trace event can be set for a variable or I/O register on a line whose background color in the event area is grayed (indicating that the line cannot be converted into a corresponding address). To set this trace event, select a variable or I/O register on a line whose background color in the event area is white.**
- 3. The types of access that can be set from the source text or disassembled text are only a read/write. To change the access type to a read or write, after setting trace start and end events, edit them in a dialog box that opens from the [Events panel](#) (see “[\(4\) Editing trace start and trace end events](#)”).**

- To set events for registered watch expressions

Perform this operation on the [Watch panel](#).

After selecting the watch expression for which you want to set an event (multiple selections not accepted), perform the following operation from the context menu. However, the watch expressions you can select are only global variables, static variables in function, static variables in file, and I/O registers.

Note that when you have performed this operation, it is interrupted as if a trace event (access-related) has been set for the selected watch expression, and it is managed on the [Events panel](#) (see “[2.17 Event Management](#)”).

Access type	Operation
Read/write	<p>[Trace start event]</p> <p>Choose [Record Start R/W Value] from [Trace Output] and then hit the [Enter] key.</p> <p>[Trace end event]</p> <p>Choose [Record End R/W Value] from [Trace Output] and then hit the [Enter] key.</p> <p>If a value is specified in the text box of the menu at this time, collection of trace data is started or finished only when a read/write is performed with a specified value. If no values are specified, collection of trace data is started or finished when a selected watch expression is read or written to, no matter what value is read or written.</p>

Cautions 1. Only the watch expressions in current scope are selectable.

To use any watch expression outside the current scope for a trace event, select a watch expression that has its scope specified.

2. The access types that can be set from watch expressions are only a read/write. To change the access type to a read or write, after setting trace start and end events, edit them in a dialog box that opens from the [Events panel](#) (see “[\(4\) Editing trace start and trace end events](#)”).

When a trace start event and trace end event are set, they are managed collectively on the [Events panel](#) as one instance of a trace event. (When you click the "+" mark at a trace event item, detailed information on the trace start and trace end events you've set is displayed.)


Cautions 1. When both the trace start and trace end events have been set, trace data is collected every time the start condition and end condition are met. However, only the trace data that were acquired immediately before the program stopped are actually displayed in the [Trace panel](#).

2. [E20(JTAG) [RX600 Series]]

Do not use the trace start and trace end events if you have enabled [Real-time RAM monitor] in the [\[Usage of trace function\]](#) property under the [\[Trace\]](#) category. Also delete any events using the [Events panel](#) that have been set.

Remarks 1. If either one of the trace start and trace end events set is [Enabled](#), the unconditional trace event checkbox on the [Events panel](#) is automatically deselected, so that collection of trace data linked to the start of program execution is not performed. (The tracer is not actuated until the condition for the set trace start event becomes true.)

2. The trace end event, if unnecessary, may be left unset.
3. The event mark varies with the set states of events (see “[2.17.1 Changing states of setting \(Enabled/Disabled\)](#)”).

Also, if a new event is set at a place where an event has already been set, the event mark  is displayed, indicating that multiple events have been set.

4. [Simulator]

If either one of the trace start and trace end events set is [Enabled](#), the [Use trace function] property

in the [\[Trace\]](#) category on the [Property panel's \[Debug Tool Settings\] tab](#) has its specification automatically changed to [Yes], with the trace function enabled.

(2) Combining multiple events [E1] [E20]

If there are two or more events set for the trace start event, collection of trace data can be made to start when the trace start event satisfies the following combination conditions.

Also, if there are two or more events set for the trace end event, collection of trace data is finished when the condition for any one of trace end events is met.

Table 2-14. Combination Conditions for the Trace Start Event

Combination condition	Description
OR	Collection of trace data begins when any one of the trace start events set occurs (i.e., its designated condition is met).
AND	Collection of trace data begins when all of the trace start events set occur, irrespective of time base.
Sequential	Collection of trace data begins when the trace start events set occur in a specified order. Also, an event, but only one, of trace start events can be registered as a reset event (R event). When a registered event occurs, the other trace start events that have had occurred up to that point of time are all cleared.

To edit a set trace event, choose [Event] from the [View] menu, select the trace event on the [Events panel](#) that is opened, and then click [Edit Condition ...] on the context menu.

Do your editing in a dialog box that is opened by this operation.

For details on how to edit in a dialog box, see “[\(3\) Editing combination conditions of events \[E1\] \[E20\]](#)”.

- Cautions 1.** When the combination conditions are sequential, access-related events can be specified at up to the third place in the sequence.
- 2.** When the combination conditions are sequential, access-related events that have had address range conditions set can be specified at only the first place in the sequence.
- 3.** The combination condition specifiable for a trace end condition is only OR.

(3) Executing the program

Execute the program (see “[2.9 Execute Programs](#)”).

Collection of trace data is started or finished when the condition set for a trace start event or trace end event is met.

For details on how to check the collected trace data, see “[2.13.5 Displaying an execution history](#)”.

(4) Editing trace start and trace end events

To edit a trace start or trace end event you've set, choose [Event] from the [View] menu, select the event displayed in detailed information of trace on the [Events panel](#) that is opened, and then click [Edit Condition ...] on the context menu.


Do your editing in a dialog box that is opened by this operation.

For details on how to edit in an execution-related event dialog box, see “[\(1\) Editing execution-related events](#)”.

Also, for details on how to edit in an access-related event dialog box, see “[\(2\) Editing access-related events](#)”.

(5) Deleting a trace start or trace end event

To delete a trace start or trace end event you've set, right-click the event mark in the event area and select [Delete Event] from the context menu that is displayed.

Also, there is another way to delete a set event. Choose [Event] from the [View] menu, select the trace event you want to delete on the [Events panel](#) that is opened, and then click the  button in the toolbar. (See “[2.17.5 Deleting events](#)”.)

Caution If either a trace start or trace end event is deleted from the event marks on the event area, all of the corresponding event marks are deleted.

2.13.4 Collecting an execution history only when conditions are met

It is possible to collect an execution history of the program only when some conditions are met.

(1) When an access to a variable or I/O register occurred

In cases where a point trace event is set, when and only when a specified access is made to any variable or I/O register, information on that is collected as trace data.

Follow the procedure described below to set a point trace event.

Cautions 1. Regarding settings of point trace events (e.g., limitations on the number of valid events), also see “[2.17.7 Points to note regarding event setting](#)”.

2. [Simulator]

For string-manipulating and multiply-and-accumulate instructions, only the last data access is the subject to be checked for event.

Remarks 1. Accesses by DMAC (Direct Memory Access Controller) and DTC (Data Transfer Controller) are not supported.

2. [Simulator]

If either one of the point trace events set is [Enabled](#), the [\[Usage of trace function\]](#) property in the [\[Trace\]](#) category on the [Property panel](#)'s [\[Debug Tool Settings\]](#) tab has its specification automatically changed to [Yes], with the trace function enabled.

(a) For access to a variable or I/O register in the source text or disassembled text

Perform this operation in the [Editor panel/Disassemble panel](#) in which the source text/disassembly text is displayed.

Select the variable or I/O register as the subject to access and, according to the access type you specify, perform the following operation from the context menu. However, the variables you can select are only global variables, static variables in function, or static variables in file.

Note that when you have performed this operation, it is interrupted as if a point trace event has been set for the selected variable or I/O register, and it is managed on the [Events panel](#) (see “[2.17 Event Management](#)”).

Access Type	Operation
Read/Write	Select [Record Reading Value] from [Trace Settings].
Write	Select [Record Writing Value] from [Trace Settings].
Read/write	Select [Record R/W Value] from [Trace Settings].

Remark Only the variables in current scope are selectable.

(b) For access to a registered watch expression

Perform this operation on the [Watch panel](#).

Select the watch expression as the subject to access and perform the following operation from the context menu. (See “[2.11.6 Displaying and changing watch expressions](#)”.)

However, the watch expressions you can select are only global variables, static variables in function, static variables in file, and I/O registers.

Note that when you have performed this operation, it is interrupted as if a point trace event has been set for the selected watch expression, and it is managed on the [Events panel](#) (see "2.17 Event Management" for details).

Access Type	Operation
Read	Select [Record Reading Value] from [Trace Output].
Write	Select [Record Writing Value] from [Trace Output].
Read/write	Select [Record R/W Value] from [Trace Output].


Remark Only the watch expressions in current scope are selectable.

To use any watch expression outside the current scope for a point trace event, select a watch expression that has its scope specified.

When you've finished setting a point trace event, execute the program (see "2.9 Execute Programs").

When the condition for the point trace event you've set is met during program execution, information on that is collected as trace data. For details on how to check trace data, see "2.13.5 Displaying an execution history".

If you want to edit the point trace event you've set, choose [Event] from the [View] menu, select the event name you want to edit on the [Events panel](#) that is opened, and then click [Edit Condition ...] on the context menu. (See "(2) Editing access-related events".)

Also, if you want to delete the point trace event you've set, choose [Event] from the [View] menu, select the event name you want to delete on the [Events panel](#) that is opened, and then click  in the toolbar of this panel. (See "2.17 Event Management".)

2.13.5 Displaying an execution history

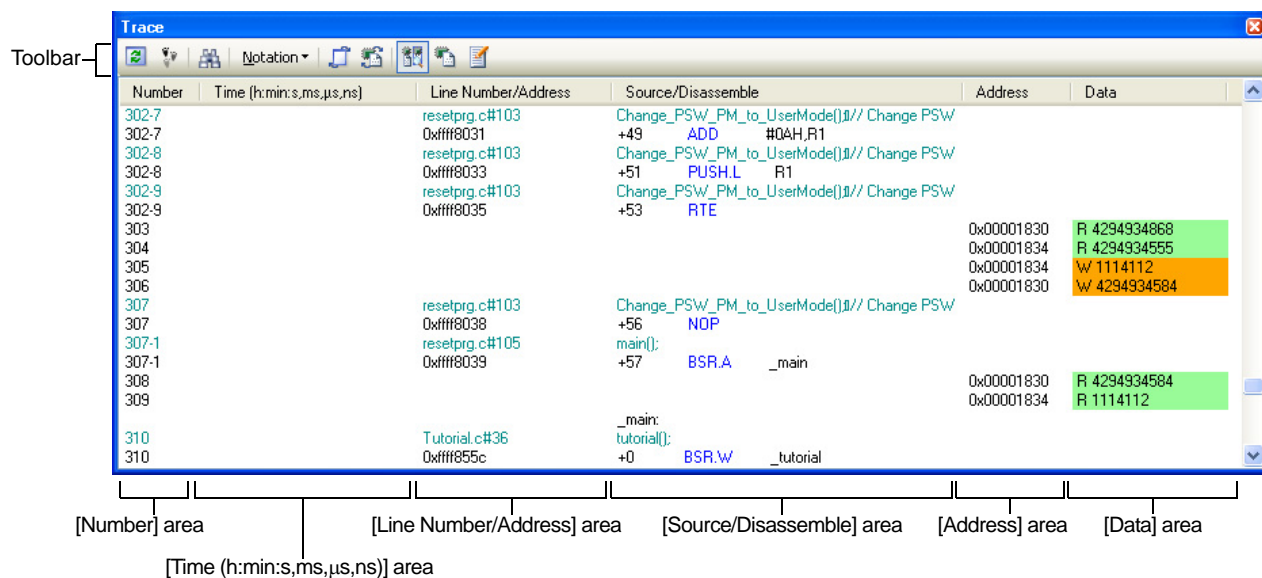
To display the collected trace data, use the [Trace panel](#) shown below.

Choose [Trace] from the [View] menu.

The trace data is displayed, by default, in a disassembled text and source text mixed mode. By selecting the appropriate [Display mode](#), it is possible to display either of the two.

For details on how to read each area and about their functionality, see the section where the [Trace panel](#) is described.

Figure 2-122. Displaying Trace Data (Trace Panel)



This section describes the following.

- (1) Changing the display mode
- (2) Changing the form in which values are displayed
- (3) Getting linked to other panels

(1) Changing the display mode

By clicking one of the toolbar buttons shown below, it is possible to change the display mode as suitable for the purpose of use.

However, these buttons are disabled while the trace function is in operation.

Table 2-15. Trace Panel Display Modes




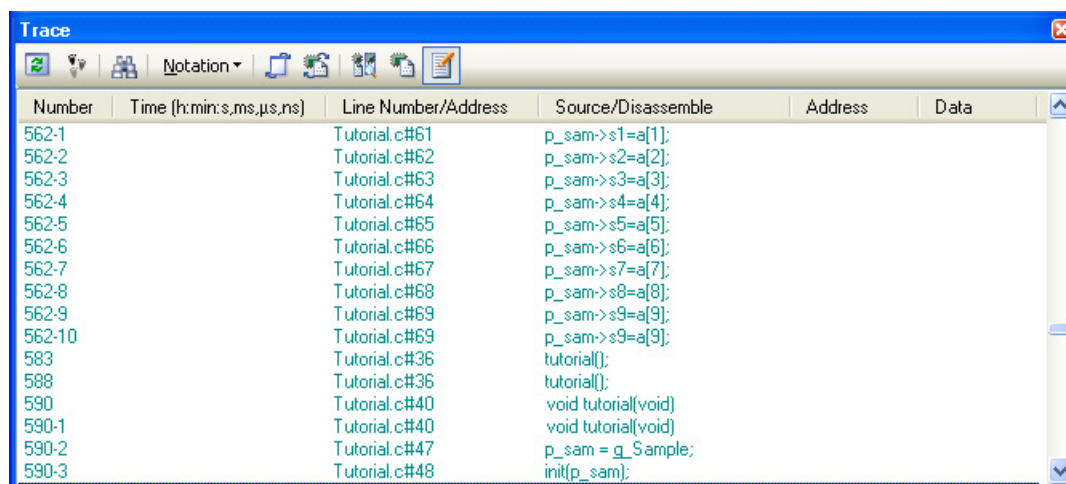
Button	Display Mode	Display Contents
	Mixed display mode	Instructions (disassemble), label names, source text (corresponding source lines), and point trace results are displayed (default).
	Disassemble display mode	Instructions (disassemble), label names, and point trace results are displayed.
	Source display mode	Source text (corresponding source lines) are displayed. However, if any place that has no debug information is executed, a notice "<No Debug Information>" is displayed.





Figure 2-123. Example of Source Display Mode (Trace Panel)



(2) Changing the form in which values are displayed

The form in which values are displayed in the [Line Number/ Address], [Address], and [Data] areas can be freely changed by using the toolbar buttons shown below.


However, these buttons are disabled during program execution.

Notation	Shows the following buttons that change the form in which values are displayed.
	Displays values on this panel in hexadecimal (default).
	Displays values on this panel in decimal.
	Displays values on this panel in octal.
	Displays values on this panel in binary.

(3) Getting linked to other panels

With the address of the currently selected line referenced as a pointer, it is possible to have the corresponding place simultaneously displayed on other panels (the focus not moved).


Clicking the toolbar button  starts a linked display on the [Editor panel](#)

Clicking the toolbar button  starts a linked display on the [Disassemble panel](#).

Clicking the button again stops the linked display.

Remark When you select [Jump to Source] or [Jump to Disassemble] from the context menu, the [Editor panel](#) or [Disassemble panel](#) opens, with the caret on it moved to the source line or address corresponding to the address of the currently selected line (the focus moved).

2.13.6 Clearing the trace memory


To clear the content of the collected trace data, click the toolbar button .

However, this button is disabled while the trace function is in operation.

Remark [Simulator]

If you've selected [Yes] for the [\[Clear trace memory before running\]](#) property in the [\[Trace\]](#) category on the [Property panel](#)'s [\[Debug Tool Settings\]](#) tab, the trace memory is cleared each time the program is run.

2.13.7 Searching for trace data

To search for the collected trace data, use the [Trace Search dialog box](#) that is opened by clicking the toolbar button . (Note that the searching function is disabled during program execution.)

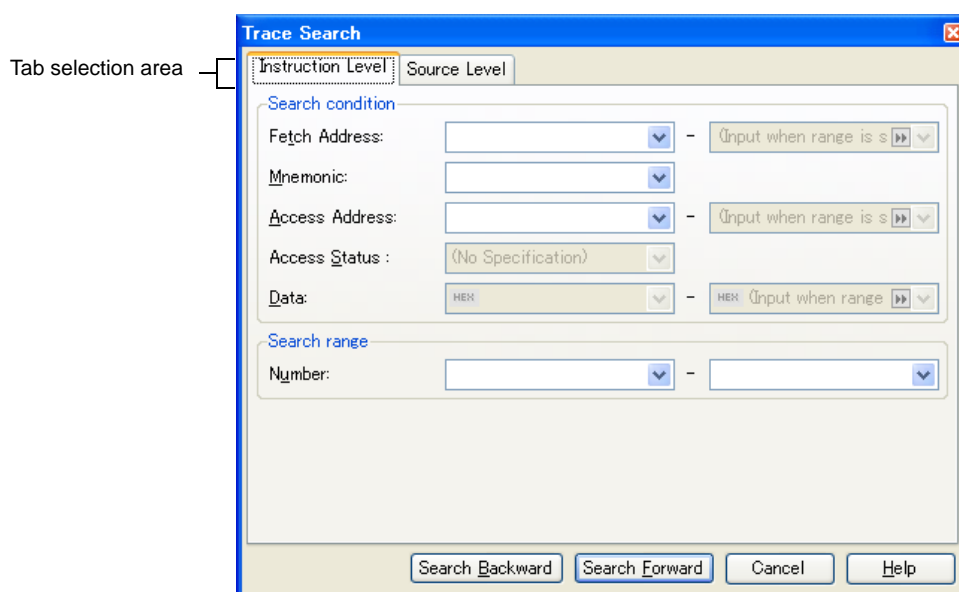
In this dialog box, perform the following operation.

Note that by selecting the appropriate tab in this dialog box, you can choose to search for trace data at the instruction level or search at the source level.

However, when you search for trace data at the instruction level, be sure that the [Trace panel](#) is displayed in the [Mixed display mode](#) or [Disassemble display mode](#).

Also, if you perform a search at the source level, be sure that the trace panel is displayed in the [Mixed display mode](#) or [Source display mode](#).

Figure 2-124. Searching for Trace Data (Trace Search Dialog Box)



This section describes the following.

- (1) [Searching at the instruction level](#)
- (2) [Searching at the source level](#)

(1) Searching at the instruction level

Search for trace data at the instruction level.

After selecting the [\[Instruction Level\]](#) tab, follow the procedure below to perform a search.

Figure 2-125. Searching for Trace Data at Instruction Level

The screenshot shows the 'Trace Search' dialog box with the 'Instruction Level' tab active. The 'Search condition' section contains the following fields and options:

- Fetch Address:** A text box with a dropdown arrow and a range specification button labeled 'Input when range is s'.
- Mnemonic:** A text box with a dropdown arrow.
- Access Address:** A text box with a dropdown arrow and a range specification button labeled 'Input when range is s'.
- Access Status:** A dropdown menu with '(No Specification)' selected.
- Data:** A dropdown menu with 'HEX' selected and a range specification button labeled 'HEX Input when range'.

The 'Search range' section contains a 'Number' field with a dropdown arrow and a range specification button.

At the bottom of the dialog are four buttons: 'Search Backward', 'Search Forward', 'Cancel', and 'Help'.

(a) Specifying the [Fetch Address]

Specify a fetch address, if needed as the search condition.

Enter an address expression directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The fetch address can also be specified as a range of addresses. In this case, enter address values in both the right and left text boxes to specify a range.

If the right-side text box is blank or labeled "(Input when range is specified)", the fixed address specified in the left-side text box is used to perform a search.

Note that if any address value greater than the microcontroller's address space is specified, the high-order address value is masked when a search is performed.

Also, no address values can be specified that are greater than the value representable by 32 bits.

(b) Specify [Mnemonic]

Specify a character string of instruction, if needed as the search condition.

The character string specified here is searched from within the [\[Source/Disassemble\] area](#) of the [Trace panel](#).

Enter an instruction directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-insensitive, and partial matches are also allowed.

(c) Specifying the [Access Address]

Specify an access address, if needed as the search condition.

Enter an address value in hexadecimal directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The access address can also be specified as a range of addresses. In this case, enter address values in both the right and left text boxes to specify a range.

If the right-side text box is blank or labeled "(Input when range is specified)", the fixed address specified in the left-side text box is used to perform a search.

Note that if any address value greater than the microcontroller's address space is specified, the high-order address value is masked when a search is performed.

Also, no address values can be specified that are greater than the value representable by 32 bits.

(d) Specifying the [Access Status]

This item is enabled only when [Access Address] is specified (see "[Specifying the \[Access Address\]](#)").

Select the type of access (Read/Write, Read, Write, Vector Read and DMA) from the drop-down list.

If you do not limit the type of access, select "(No specification)".

Caution The types of access for which data can be collected on the [Trace panel](#) are only Read/write, Read, and Write. Therefore, do not select Vector Read or DMA from the drop-down list.

(e) Specifying the [Data]

This item is enabled only when [Access Address] is specified (see "[Specifying the \[Access Address\]](#)").

Specify an accessed numeric value.

Enter a hexadecimal value directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The numeric value can be specified as a range of values. In this case, enter data in both the right and left text boxes to specify a range.

If the right-side text box is blank or labeled "(Input when range is specified)," the fixed numeric value specified in the left-side text box is used to perform a search.

(f) Specifying the [Number]

Specify a range of trace data to search by numbers displayed in the [\[Number\] area](#) of the [Trace panel](#).

Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "last number" are specified.)

Enter a number in decimal notation directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

If the left-side text box is blank, the number "0" is assumed.

If the right-side text box is blank, the "last number" is assumed.

(g) Clicking the [Search Backward] or [Search Forward] button

When you click the [Search Backward] button, a search is performed in the direction toward smaller numbers, with the searched spot put in selected state on the [Trace panel](#).

When you click the [Search Forward] button, a search is performed in the direction toward larger numbers, with the searched spot put in selected state on the [Trace panel](#).

(2) Searching at the source level

Search for trace data at the source level.

Select the [\[Source Level\]](#) tab.

Figure 2-126. Search for Trace Data at Source Level

The screenshot shows the 'Trace Search' dialog box with the 'Source Level' tab selected. The dialog is divided into three main sections: 'Search object', 'Search condition', and 'Search range'. In the 'Search object' section, the first radio button is selected: 'The execution part is retrieved specifying the source line'. The 'Search condition' section contains several input fields: 'Source and Line' (a text box with a drop-down arrow), 'Function Name' (a text box with a drop-down arrow), 'Variable Name' (a text box with a drop-down arrow), 'Kind' (a drop-down menu showing 'Reference/Substitution'), and 'Value' (two text boxes with 'HEX' selected in the first, separated by a minus sign). The 'Search range' section has a 'Number' field with two text boxes and drop-down arrows. At the bottom are four buttons: 'Search Backward', 'Search Forward', 'Cancel', and 'Help'.

(a) To specify a source line before performing a search (default)

Select [The execution part is retrieved specifying the source line] in the [\[Search object\]](#) area, then perform the following operation.

<1> Specifying the [Source and Line]

The character string specified here is searched from within the [\[Line/Address\]](#) area of the [Trace panel](#).

Enter a character string included in the source line you want to search directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-insensitive, and partial matches are also allowed.

- Examples 1.**
1. main.c#40
 2. main.c
 3. main

<2> Specifying the [Number]

Specify a range of trace data to search by numbers displayed in the [\[Number\]](#) area of the [Trace panel](#).

Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "*last number*" are specified.)

Enter a number in decimal notation directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

If the left-side text box is blank, the number "0" is assumed.

If the right-side text box is blank, the "*last number*" is assumed.

<3> Clicking the [Search Backward] or [Search Forward] button

When you click the [Search Backward] button, a search is performed in the direction toward smaller numbers, with the searched spot put in selected state on the [Trace panel](#).

When you click the [Search Forward] button, a search is performed in the direction toward larger numbers, with the searched spot put in selected state on the [Trace panel](#).

(b) To specify a function name before performing a search

Select [The execution part is retrieved specifying the function] in the [\[Search object\] area](#), then perform the following operation.

<1> Specifying the [Function Name]

Enter a function name you want to search directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-sensitive, and only perfect matches are allowed.

<2> Specifying the [Number]

Specify a range of trace data to search by numbers displayed in the [\[Number\] area](#) of the [Trace panel](#).

Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "*last number*" are specified.)

Enter a number in decimal notation directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

If the left-side text box is blank, the number "0" is assumed.

If the right-side text box is blank, the "*last number*" is assumed.

<3> Clicking the [Search Backward] or [Search Forward] button

When you click the [Search Backward] button, a search is performed in the direction toward smaller numbers, with the searched spot put in selected state on the [Trace panel](#).

When you click the [Search Forward] button, a search is performed in the direction toward larger numbers, with the searched spot put in selected state on the [Trace panel](#).

(c) To specify a global variable name before performing a search

Select [The execution part is retrieved specifying the global variable] in the [\[Search object\] area](#), then perform the following operation.

<1> Specifying the [Variable Name]

Enter a variable name you want to search directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-sensitive, and only perfect matches are allowed.

<2> Specifying the [Kind]

Select the type of access ([Reference/Substitution] (default), [Reference], or [Substitution]) from the drop-down list.

<3> Specifying the [Value]

Enter an accessed variable value directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The variable value can be specified as a range of values. In this case, enter data in both the right and left text boxes to specify a range.

If the right-side text box is blank, the fixed variable value specified in the left-side text box is used to perform a search.

<4> Specifying the [Number]

Specify a range of trace data to search by numbers displayed in the [Number] area of the [Trace panel](#). Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "*last number*" are specified.)

Enter a number in decimal notation directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

If the left-side text box is blank, the number "0" is assumed.

If the right-side text box is blank, the "*last number*" is assumed.

<5> Clicking the [Search Backward] or [Search Forward] button

When you click the [Search Backward] button, a search is performed in the direction toward smaller numbers, with the searched spot put in selected state on the [Trace panel](#).

When you click the [Search Forward] button, a search is performed in the direction toward larger numbers, with the searched spot put in selected state on the [Trace panel](#).

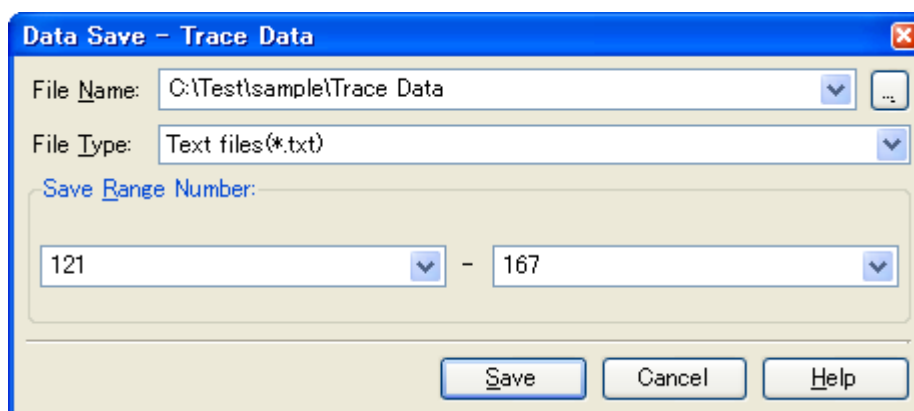
2.13.8 Saving the displayed content of an execution history

The contents of collected trace data can be saved, after specifying a range of data, in a text file (*.txt) or CSV file (*.csv) format. When saving to a file, the latest information will be retrieved from the debug tool, and saved according to the display format on this panel.

Choose [Save Trace Data As...] from the [File] menu, and the [Data Save dialog box](#) shown below is opened.

In this dialog box, follow the procedure described below to save the data.

Figure 2-127. Saving an Execution History (Data Save Dialog Box)

**(1) Specifying the [File Name]**

Specify a file name in which you want to save.

Enter it directly in the text box (specifiable in up to 259 characters) or select an input history item from the drop-down list (up to 10 history entries).

Also, you can select a file from the [Select Data Save File dialog box](#) that is opened by clicking the [...] button.

(2) Specifying the [File Type]

Select the type of file in which you want to save from the drop-down list below.

The selectable file types are as follows:

List display	Format
Text files (*.txt)	Text format (default)
CSV (Comma-Separated Variables)(*.csv)	CSV format ^{Note}

Note The data is saved with entries separated by commas (.).
If the data contains commas, each entry is surrounded by double quotes (" ") in order to avoid illegal formatting.

(3) Specifying the [Save Range Number]

Specify the "start trace number" and "end trace number" to set a range of data to be saved in a file.

Enter numeric values in decimal notation directly in the respective text boxes or select an input history item from the drop-down list (up to 10 history entries).

If you want to save all trace data, select [All Trace Data] in the drop-down list on the left side (right-side text box disabled).

Note that if a range is selected on panel, this selected range is specified, by default, in the text boxes. If no range is selected, the currently displayed range on panel is specified.

(4) Clicking the [Save] button

Trace data is saved in the specified format to a specified file.

Figure 2-128. Trace Data Output Image When Saved

Number	Time	Line Number/Address	Source/Disassemble	Address	Data
-----	-----	-----	-----	-----	-----
<i>Number</i>	<i>Time</i>	<i>Line Number/Address</i>	<i>Source/Disassemble</i>	<i>Address</i>	<i>Data</i>
:	:	:	:	:	:

2.14 Measuring the Execution Time

This section describes how to measure a program's execution time.

2.14.1 Setting the timer measurement operation [E1] [E20]

For timer measurements to be performed, it is necessary to make the following settings related to timer measurements.

Do these settings in the [\[Timer\] \[E1\]\[E20\]](#) category on the [Property panel's \[Debug Tool Settings\] tab](#).

Note that properties in this category can only be changed while CubeSuite+ is disconnected from the debug tool.

Figure 2-129. [Timer] Category [RX600 Series]

Timer	
Use 64bit counter	No
Operating frequency[MHz]	25.0000

Figure 2-130. [Timer] Category [RX200 Series]

Timer	
Operating frequency[MHz]	25.0000

(1) [Use 64bit counter] [RX600 Series]

Specify whether you want the measurement counter to be used in 32 bits x 2 or in 64 bits x 1.

When you select [Yes], a 64-bit measurement counter can be used, in which case, however, measurements can be taken in only one section.

Caution [RX200 Series]

This property is not displayed because its measurement counter is 24 bits x 1 only.

(2) [Operating frequency[MHz]]

Specify the counter's operating frequency that is referenced when converting count values into time.

Enter a numeric value directly in the range 0.0001 to 999.999 (in MHz units) to specify it.

If no operating frequency is specified, count values are displayed.

- Cautions 1.** The result of timer measurement event is calculated based on the operating frequency and count values entered in this Property panel. Therefore, using a program that switches operating frequencies while the program is running will result in incorrect measurement result.
- 2.** Timer measurement proceeds even if the microcontroller is reset. However, the results may not be correct because the clock settings for measurement have been initialized.

2.14.2 Measuring execution time from start to stop

The debug tool has a preinstalled function to measure a program's execution time from start to stop (Run-Break time).

Therefore, when a program starts running, its execution time is automatically measured.

The measurement result can be confirmed by one of the following methods.

- (1) [Checking the status bar for confirmation](#)
- (2) [Checking the event panel for confirmation](#)

- Cautions 1.** The measurement result of the Run-Break timer event includes the time between the start of the measurement and the program execution as well as the time between the program break and

the measurement break. Note that this result is used only for a reference purpose as it contains clock error of the measurement clock source.

2. The execution time displayed in the **Status bar** and the **Events panel** is the time measured while the program is being executed. Values smaller than 100 μ s are discarded. The execution time will be incorrect when Step in, Step over or Return out is performed.

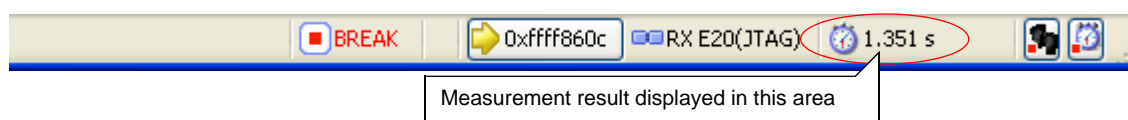
Remark This function is actuated by a Run-Break timer event, which is one of the built-in events set in the debug tool by default.

The Run-Break timer event is always **Enabled** (settings not changeable).

(1) Checking the status bar for confirmation

When a program has stopped running, the measurement result is displayed in the status bar on the **Main window** (While no measurements are made, this status shows "Not measured").

Figure 2-131. Example of a Run-Break Timer Event Measurement Result (Status Bar)



(2) Checking the event panel for confirmation

When a program has stopped running, the measurement result is displayed as a Run-Break timer event on the **Events panel** that is opened by selecting [Event] from the [View] menu.

Figure 2-132. Example of a Run-Break Timer Event Measurement Result (Event Panel) [E1] [E20]

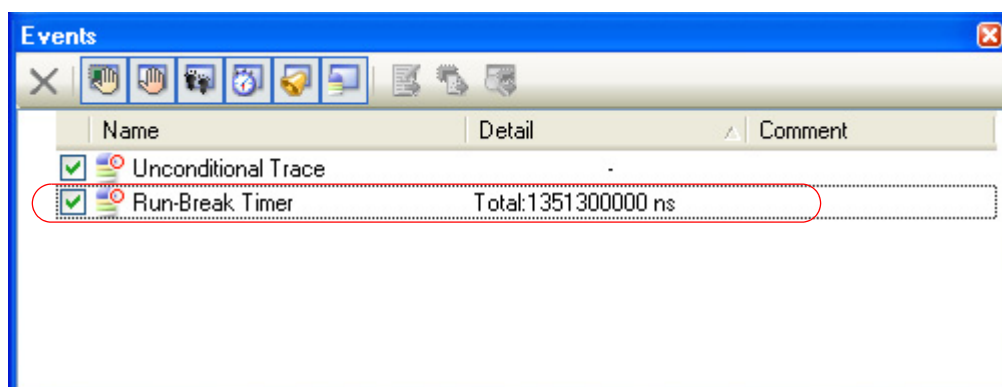
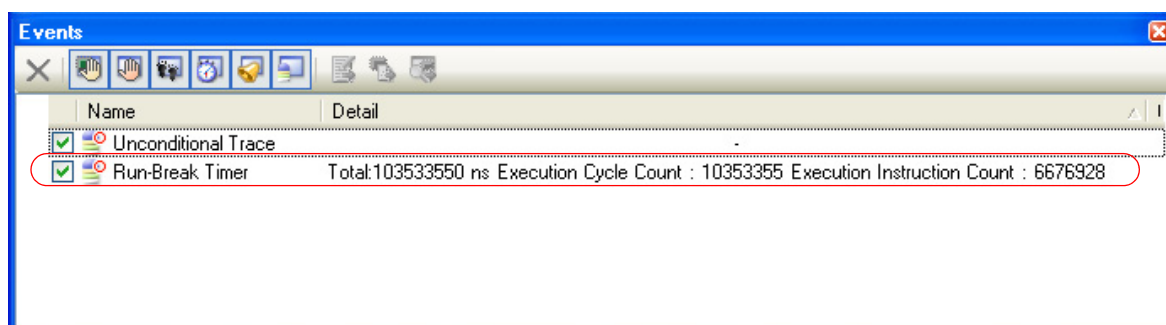


Figure 2-133. Example of a Run-Break Timer Event Measurement Result (Event Panel) [Simulator]



2.14.3 Measuring execution time in a section [E1] [E20]

By setting timer measurement events (timer start event and timer end event) it is possible to measure a program's execution time in any section in its execution process.

To use this function, follow the procedure described below.

- (1) [Setting the timer start event and timer end event](#)
- (2) [Execute the program](#)
- (3) [Editing a timer measurement event](#)
- (4) [Deleting a timer start event or timer end event](#)

Cautions 1. [Simulator]

Timer measurement events are not supported.

2. Regarding settings of timer measurement events (e.g., about limitation on valid number of events), also see "[2.17.7 Points to note regarding event setting](#)".

(1) Setting the timer start event and timer end event

On the [Editor panel](#), [Disassemble panel](#) or [Watch panel](#), set the events at which you want a timer measurement to start and end.

(a) How to set a timer start event

- To set from the [Editor panel](#)

Move the caret to the line or address **Note 1** at which you want a timer measurement to start and then select [Timer Settings] on context menu and from it, choose [Start Timer].

A timer start event is set for the instruction at the beginning address corresponding to the line or address at the caret position.

- To set from the [Disassemble panel](#)

Move the caret to the line or address **Note 1** at which you want a timer measurement to start and then select [Timer Settings] on context menu and from it, choose [Start Timer] and then [Set Timer * **Note 2**].

A timer start event is set for the instruction at the beginning address corresponding to the line or address at the caret position.

Move the caret to the variable at which you want a timer measurement to start and then select [Timer Settings] on context menu and from it, choose [Set Timer Start R/W Value] and then [Set Timer * **Note 2**].

A timer start event is set for the variable at the caret position.

- To set from the [Watch panel](#)

Move the caret to the watch expression **Note 3** at which you want a timer measurement to start and then select [Timer Settings] on context menu and from it, choose [Set Timer Start R/W Value] and then [Set Timer * **Note 2**].

A timer start event is set for the watch expression on the watch panel.

(b) How to set a timer end event

- To set from the [Editor panel](#)

Move the caret to the line or address **Note 1** at which you want a timer measurement to finish and then select [Timer Settings] on context menu and from it, choose [End Timer].

A timer end event is set for the instruction at the beginning address corresponding to the line or address at the caret position.

- To set from the [Disassemble panel](#)

Move the caret to the line or address **Note 1** at which you want a timer measurement to finish and then select [Timer Settings] on context menu and from it, choose [End Timer] and then [Set Timer * **Note 2**]

A timer end event is set for the instruction at the beginning address corresponding to the line or address at the caret position.

Move the caret to the variable at which you want a timer measurement to finish and then select [Timer Settings] on context menu and from it, choose [Set Timer End R/W Value] and then [Set Timer * **Note 2**].

A timer end event is set for the variable at the caret position.

- To set from the [Watch panel](#)

Move the caret to the watch expression **Note 3** at which you want a timer measurement to finish and then select [Timer Settings] on context menu and from it, choose [Set Timer End R/W Value] and then [Set Timer * **Note 2**].

A timer end event is set for the watch expression on the [Watch panel](#).

Notes 1. Timer start event/timer end event cannot be set to lines with no address indication.

2. The asterisk (*) in the menu [Set Timer *] denotes a channel number as a number for a timer measurement section. To set timer start and end events in one section, be sure to select the same channel number.

Note that the selectable channel numbers vary with each microcontroller used and depend on how the [\[Timer\] \[E1\]\[E20\]](#) category on the [Property panel's \[Debug Tool Settings\] tab](#) is set, as shown below.

Microcontroller	[Use 64bit counter] property	Function
RX600 Series	No	Specifiable from 2 sections (32-bit), [Set Timer 1] and [Set Timer 2]
	Yes	Only 1 section (64-bit), [Set Timer 1], is specifiable
RX200 Series	-	Only 1 section (24-bit), [Set Timer 1], is specifiable

3. This watch expression can only be a global variable, static variable in function, static variable in file, and an IO register.
4. Measurement by the timer is possible even if either a timer start or timer stop event is set. When only a timer start event is set, measurement ends as soon as the program stops running. When only a timer stop event is set, measurement starts as soon as the program starts running.

When timer start and end events are set, the following event marks are displayed in the event area of the relevant line, address, or watch expression.

Also, on the [Events panel](#), they are managed collectively as one instance of a timer measurement event. (By clicking the "+" mark at a timer measurement event item, it is possible to check information on the timer start and end events you've set.

Table 2-16. Event Marks of the Timer Start and End Events



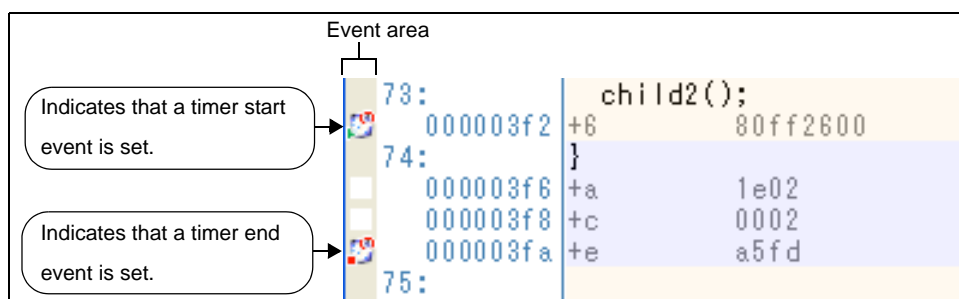
Type	Event mark
Timer start	
Timer end	

Figure 2-134. Example of Timer Start and End Events Set

**Caution [RX600 Series]**

The timer measurement will be suspended when the timer start event occurs twice even though the timer end event has not occurred. Measurement will be resumed on the next occurrence of the time start event.

Remark The event mark differs depending on how an event is set (see "2.17.1 Changing states of setting (Enabled/Disabled)").

Also, if a new event is set at a place where an event has already been set, the event mark (🔒) is displayed, indicating that multiple events have been set.

(2) Execute the program

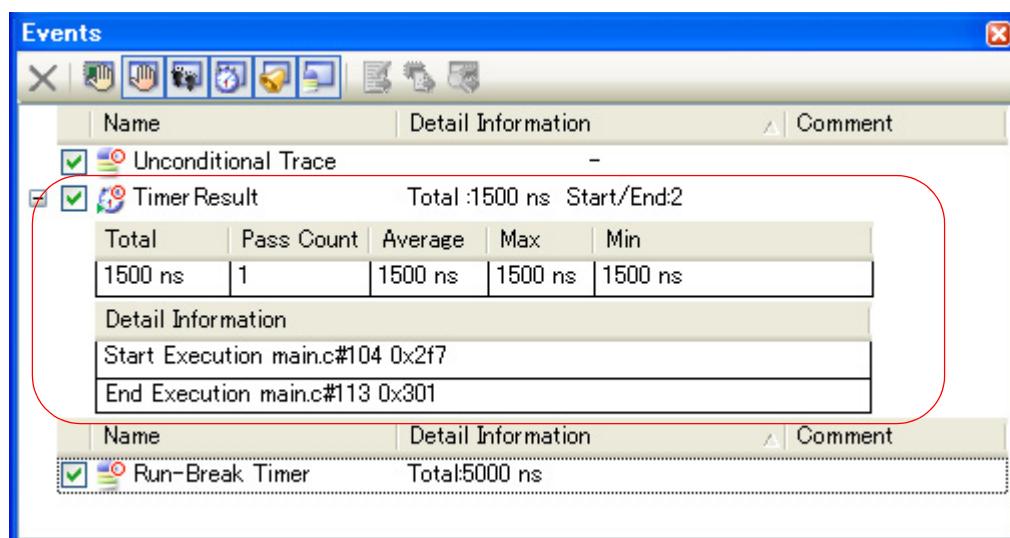
Execute the program (see "2.9 Execute Programs").

When an instruction for which a timer start event or timer end event has been set is executed, a timer measurement is started or finished.

After execution of the program has stopped, the measurement result can be confirmed as a timer measurement event on the [Events panel](#) that is opened by choosing [Event] from the [View] menu, as shown below.

Note that this timer measurement event is a particular type of event that is displayed on only the [Events panel](#) when either a timer start event or a timer end event has been set.

Figure 2-135. Example of the Result of Measurement by Timer Measurement Event (Timer Start and End Events)



(3) Editing a timer measurement event

To edit detailed information on a timer measurement event that has had start and end events set, use the [Detailed Settings of Timer Measurement dialog box \[E1\] \[E20\]](#). This dialog box is opened by selecting the event to be edited on the [Events panel](#) and then selecting [Edit Condition ...] on context menu.

(a) To edit a measurement item

In the Detailed settings of Timer Measurement dialog box, a measurement item can be specified from the following.

Measurement item	Function
Execution cycle Note	Measures the number of elapsed cycles (ICLK) in a specified section.
Execution cycle (Supervisor mode)	Measures the number of elapsed cycles (ICLK) while operating in supervisor mode.
Exception and interrupt cycle	Measures the number of cycles (ICLK) needed to process interrupts (including exceptions).
Exception cycle	Measures the number of cycles (ICLK) needed to process exceptions.
Interrupt cycle	Measures the number of cycles (ICLK) needed to process interrupts.
Execution count	Measures the number of instructions that have their execution completed.
Exception and interrupt count	Measures the number of times interrupts, including exceptions, were accepted.
Exception count	Measures the number of times exceptions occurred.
Interrupt count	Measures the number of times interrupts were accepted.

Note [RX200 Series]

Only [Execution cycle] is displayed for the measurement item.

(b) To edit an only once measurement

When you specify [Yes] for this property, the timer measurement is finished by measuring a specified section only once. If you want to measure a total number of times a specified section has been passed, be sure to specify [No].

Remark [RX600 Series]

While the measurement listed below is performed, even if the start event and end event occur, if the condition for measurement is not satisfied even once, the results of measurement will not be displayed.

- Execution count
- Exception and interrupt count
- Exception count
- Interrupt count

(4) Deleting a timer start event or timer end event

To delete a timer start event or timer end event you've set, right-click the event mark in the event area and then select [Delete Events] on the context menu that is displayed.

Also, there is another way to delete a set event. Choose [Event] from the [View] menu, select the relevant breakpoint on the ensuing [Events panel](#), and then click the (X) button in the toolbar (see "[2.17.5 Deleting events](#)").

2.14.4 Range of measurable time

There is a finite range of measurable time for timer measurements by Run-Break timer events (see "[2.14.2 Measuring execution time from start to stop](#)") or timer measurement events (see "[2.14.3 Measuring execution time in a section \[E1\] \[E20\]](#)"), as shown below.

Table 2-17. Range of Measurable Time

Debug tool	Run-Break timer event		Timer measurement event	
E1 E20 [RX600 Series]	Min	100 microsecond	Min	-
	Max	Approx. 72 hours	Max	Depends on CPU clock frequency 32-bit counter x 2 channels or 64-bit counter x 1 channel Overflow detection available
E1 E20 [RX200 Series]	Min	100 microsecond	Min	-
	Max	Approx. 72 hours	Max	Depends on CPU clock frequency 24-bit counter x 1 channel Overflow detection available
Simulator	Depends on CPU clock frequency		Depends on CPU clock frequency	

2.15 Measure Coverage [Simulator]

This section describes coverage measurements that are conducted using the coverage facility.

There are several kinds of coverage measurement methods. Of these, CubeSuite+ performs, in areas designated below, a code coverage measurement of fetch-related operations on source lines and functions (C0 coverage) and a data coverage measurement of access-related operations on variables.

The areas in which CubeSuite+ performs coverage measurements are as follows:

Table 2-18. Subject Areas of Coverage Measurements

Debug Tool	Subject area
Simulator	Internal ROM/RAM, emulation ROM/RAM

Remark C0 coverage refers to an instruction coverage rate (statement coverage).

For example, if all instructions (statements) in code are executed at least once, then C0 = 100%.

2.15.1 Configure the coverage measurement

To use the coverage facility, it is necessary to make coverage measurement-related settings in advance.

Make settings in the [\[Coverage\] \[Simulator\]](#) category on the [\[Debug Tool Settings\] tab](#) on the [Property panel](#).

Figure 2-136. [Coverage] Category [Simulator]

Coverage	
Use coverage function	Yes
Reuse coverage result	No

(1) [Use coverage function]

Specify by a drop-down list whether or not you want to use the coverage function.

To use the coverage function, select [Yes]. (By default, [No] is selected.)

(2) [Reuse coverage result]

This property is displayed only when [\[\[Use coverage function\]\]](#) property is set to [Yes].

When disconnecting from the debug tool, specify from the drop-down list whether to automatically save the acquired code coverage measurement result and reproduce it next time you connect to the debug tool.

To reproduce the contents of the last obtained code coverage measurement results, select [Yes]. (By default, [No] is selected.)

2.15.2 Display coverage measurement results

Coverage measurement starts (ends) automatically with the start (end) of the program execution.

(1) Code coverage ratio

(a) Display of code coverage rates for source lines and disassemble lines

The code coverage ratio is indicated on the [Editor panel](#) or [Disassemble panel](#) that is displaying the target program.

On each panel, the target source text lines and disassemble result lines are shown in color-coded background (see [Table 2-20.](#)) according to their code coverage ratio that was calculated based on the formula described in [Table 2-19.](#)

The results are not shown when disconnected from the debug tool or during the program execution.

Selecting [Clear coverage information] from the context menu will reset all the coverage information acquired, including the color coding on each panel.

Table 2-19. Calculating Code Coverage Ratio for Source Lines and Disassemble Lines

Panel	Calculation method
Editor panel	Number of bytes executed in the address area corresponding to the source line / Total number of bytes present in the address area corresponding to the source line
Disassemble panel	Number of bytes executed in the address area corresponding to the disassemble result line / Total number of bytes present in the address area corresponding to the disassemble result line

Table 2-20. Color-coded Code Coverage Measurement Results (Default)

Code Coverage	Background Color
100 %	Source text/ disassemble results
1 to 99 %	Source text/ disassemble results
0 % (not executed)	Source text/ disassemble results

- Remarks 1.** The code coverage measurement result displayed on each panel is updated at every program break.
- 2.** The above color coding rule is determined according to the settings in [General - Font and Color] category category in the Option dialog box.
- 3.** The above color coding rule does not apply to the lines that are outside the target area (see "Table 2-18. Subject Areas of Coverage Measurements").
- 4.** The Editor panel will not display the code coverage measurement result in cases where the source file currently displayed is updated after the update of the downloaded modules.

Figure 2-137. Example of Code Coverage Measurement Results (Editor Panel)

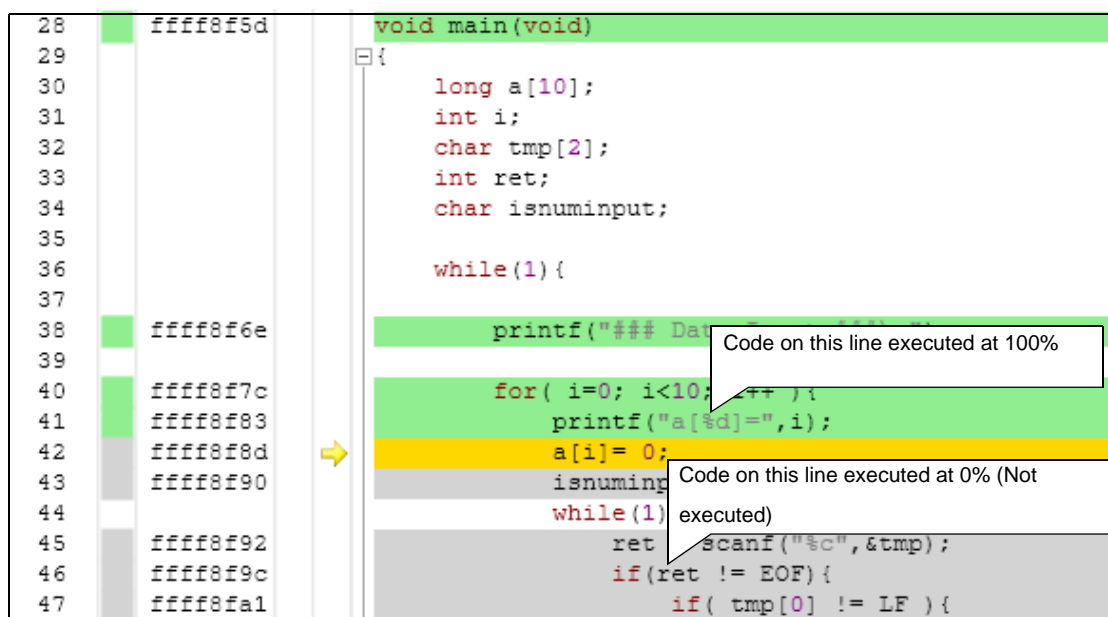


Figure 2-138. Example of the Code Coverage Measurement Result (Disassemble Panel)

	main:		
ffff8f5d	7100d4	ADD	#-2CH, R0, R0
ffff8f60	fb3274d5ffff	MOV.L	#-00002A8CH, R12
ffff8f66	ef0b	MOV.L	R0, R11
ffff8f68	fba27bd5ffff	MOV.L	#-00002A85H, R10
38:	printf("### Data Input ###")		
ffff8f6e	fb3260d5ffff	MOV.L	
ffff8f74	7ea3	PUSH.L	R3
ffff8f76	05ed0300	BSR.A	_printf
ffff8f7a	6240	ADD	#4H, R0
40:	for(i=0; i<10; i++){		
ffff8f7c	6607	MOV.L	#0H, R7
ffff8f7e	710804	ADD	#04H, R0, R8
ffff8f81	ef89	MOV.L	R8, R9
41:	printf("a[%d]=", i);		
ffff8f83	7ea7	PUSH.L	R7
ffff8f85	7eac	PUSH.L	R12
ffff8f87	05dc0300	BSR.A	_printf
ffff8f8b	6280	ADD	#8H, R0
42:	a[i]= 0;		
ffff8f8d	f89600	MOV.L	#00H, [R9]
43:	isnuminput = FALSE;		
ffff8f90	6606	MOV.L	
45:	ret = scanf("%c", &tmp);		
ffff8f92	7eab	PUSH.L	
ffff8f94	7eaa	PUSH.L	R10
ffff8f96	05fe0300	BSR.A	_scanf
ffff8f9a	6280	ADD	#8H, R0
46:	if(ret != EOF){		
ffff8f9c	7501ff	CMP	#-01H, R1
ffff8f9f	2074	BEQ.B	_main+B6H
47:	if(tmp[0] != LF){		

Code on this line executed at 100%

Code on this line executed at 0% (Not executed)

(b) Display of code coverage rates for each function

Check the [Code Coverage] item in the Function panel of the analysis tool for the code coverage ratio of each function (i.e., function coverage ratio).

For details on the "code coverage ratio of functions," see the separate edition "CubeSuite+ Analysis."

(2) Data coverage rates

Check the [Data Coverage] item in the Variable panel of the analysis tool for the data coverage ratio of each variable.

For details on the "data coverage ratio of variables," see the separate edition "CubeSuite+ Analysis."

2.16 Set an Action into Programs

This section describes how to set the specified action into the program.




2.16.1 Insert printf

Setting the Printf event as one of action events allows you to output the value of the specified variable expression to the [Output panel](#). This can be done by executing a printf command after temporarily stopping the program at an arbitrary position.

To use this function, follow the steps below.

- (1) [Set a Printf event](#)
- (2) [Execute the program](#)
- (3) [Check the output result](#)
- (4) [Edit Printf event](#)

Cautions 1. Also see "[2.17.7 Points to note regarding event setting](#)" for details on action events (e.g. limits on the number of enabled events).

- 2.** No action events occur during step execution ( /  / ) or execution ignoring break-related events (.

(1) Set a Printf event

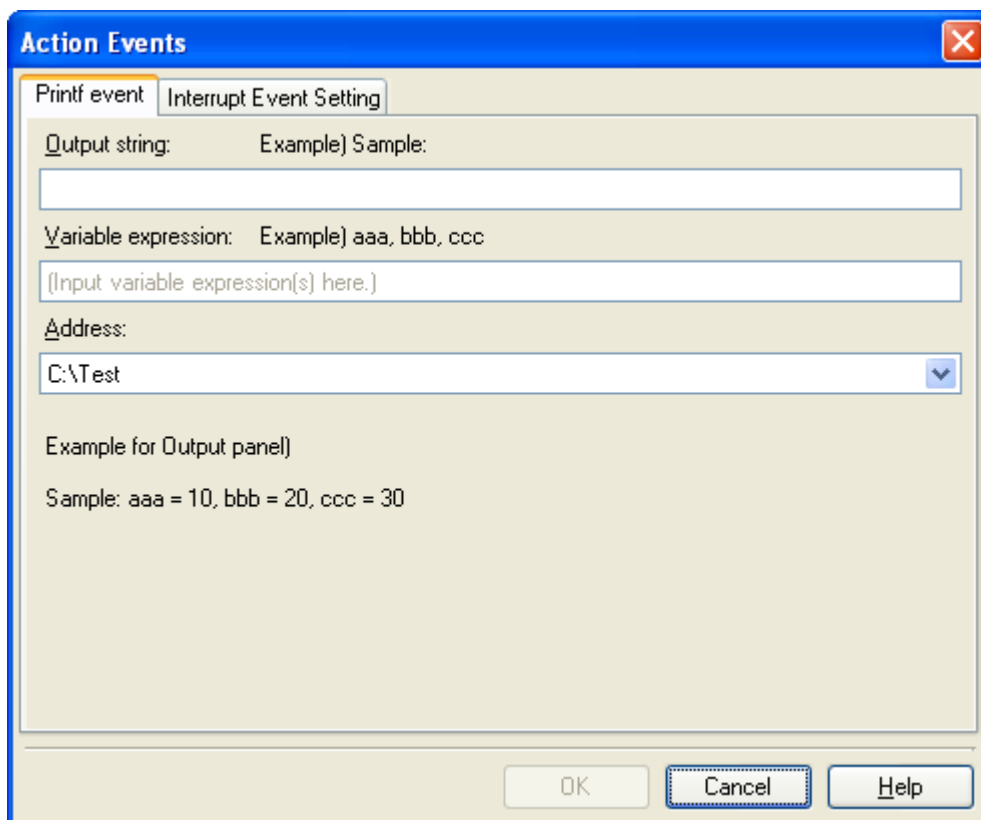
Set a Printf event to the position where you want to execute the printf command in the [Editor panel/Disassemble panel](#).

In the Editor panel/Disassemble panel, move the caret to the line/address^{Note} at which you want to set a Printf event and then select [Register Action Event...] from the context menu to open the following [Action Events dialog box](#).

In this dialog box, follow the steps below.

Note Printf events cannot be set to lines with no address indication.

Figure 2-139. Set Printf Event (Action Events Dialog Box)

**(a) Specify [Output string]**

Directly enter from the keyboard the characters to add when output to the [Output panel](#). Characters must be in one line (spaces allowed).

(b) Specify [Variable expression]

Specify the variable expression for the Printf event to take place.

Type a variable expression directly into the text box (up to 1024 characters).

You can specify up to 10 variable expressions for a single Printf event by separating them with commas ",".

If this dialog box opens with a variable expression selected in the [Editor panel/Disassemble panel](#), the selected variable expression appears as the default.

For the basic input format that can be specified as variable expressions and the values output by Printf event, see "[Table A-14. Relationship between Variable Expressions and Output Value \(Printf Event\)](#)".

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").

(c) Specify [Address]

Designate an address that specifies the Printf event.


By default, the presently specified address is displayed.

When editing, you can either type an address expression directly into the text box (up to 1024 characters), or select one from the input history from the drop-down list (up to 10 items).

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").

(d) Click the [OK] button

Set the Printf event to the line at the caret position in the [Editor panel/Disassemble panel](#).

When the Printf event is set, the  mark is displayed in the event area on the Editor panel/Disassemble panel, and the set Printf event is managed in the [Events panel](#) (see "2.17 Event Management").

(2) Execute the program

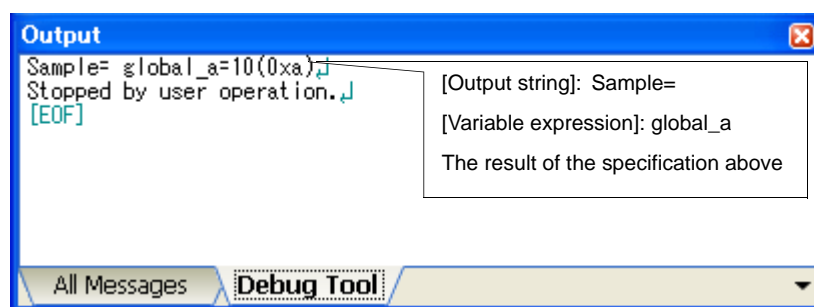
Execute the program (see "2.9 Execute Programs").

By executing the program, the program momentarily stops immediately before executing the instruction at the location where this event is set, and the value of the variable expression specified in this dialog box is output to the [Output panel](#).

(3) Check the output result

The following figure shows how the value of the specified variable expression is output to the [\[Debug Tool\]](#) tab of the [Output panel](#) (see "Figure A-44. Output Result Format of Printf Event").

Figure 2-140. Example of Output Result of Printf Event

**(4) Edit Printf event**

You can edit a Printf event which has already been set.

When editing, click the target Printf event in the [Events panel](#) and select [Edit the conditions...] from the context menu. This will open the [Action Events dialog box](#) in which you can edit the items. When finished, click [OK].

2.16.2 Insert an interrupt event [Simulator]

Setting an interrupt event as one of the action events allows you to generate an interrupt at a desired location while the program is running. To use this function, follow the steps below.

- (1) [Set an interrupt event](#)
- (2) [Execute the program](#)
- (3) [Edit interrupt event](#)

Caution Also see "2.17.7 Points to note regarding event setting" for details on action events (e.g. limits on the number of enabled events).

(1) Set an interrupt event

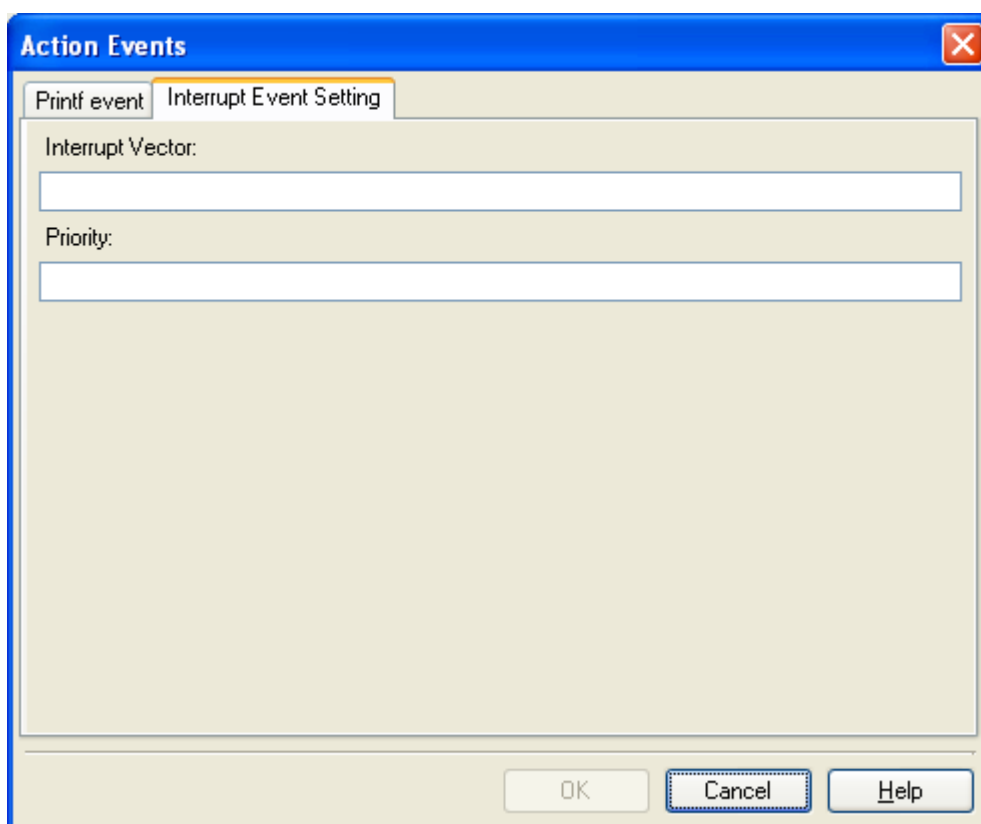
In the [Editor panel/Disassemble panel](#), set an interrupt event at a location where you want an interrupt request to be generated.

In the [Editor panel/Disassemble panel](#), move the caret to the line/address^{Note} at which you want to set an interrupt event and then select [Register Action Event...] from the context menu to open the following [Action Events dialog box](#).

In this dialog box, follow the steps below.

Note Interrupt events cannot be set to lines with no address indication.

Figure 2-141. Set Interrupt Event (Action Events Dialog Box: [Set interrupt event] Tab)



(a) Specify the [Interrupt vector]

Specify the interrupt vector by directly entering a corresponding number between 0 and 255.

(b) Specify the [Priority order]

- [RX610 Group]

Specify the priority order by directly entering a number between 1 and 8.

When 8 is specified, the event is handled as a fast interrupt.


- [Non-RX610 Group]

Specify the priority order by directly entering a number between 1 and 16.

When 16 is specified, the event is handled as a fast interrupt.

(c) Click the [OK] button

Set the interrupt event to the line at the caret position in the [Editor panel/Disassemble panel](#).

When the interrupt event is set, the  mark is displayed in the event area on the Editor panel/Disassemble panel, and the set interrupt event is managed in the [Events panel](#) (see "2.17 Event Management").

(2) Execute the program

Execute the program (see "2.9 Execute Programs").

When the program is executed, the interrupt request will occur immediately before the execution of the instruction at which the interrupt event has been set. Once the interrupt request is accepted by the CPU, interrupt exception will take place.

- Cautions**
1. Once the generated interrupt request is accepted and interrupt exception executed, the interrupt request will be cleared.
 2. If an interrupt request is made by another event while the generated interrupt request is being reserved, one with higher interrupt priority will be valid.

(3) Edit interrupt event

You can edit an interrupt event which has already been set.

When editing, click the target interrupt event in the [Events panel](#) and select [Edit the conditions...] from the context menu. This will open the [Detailed Settings of Interrupt Events dialog box \[Simulator\]](#) in which you can edit the items. When finished, click [OK].

(a) Edit [Number of passages]

Directly enter a value between 1 and 16383.

The event will be encountered when the event conditions meeting the number of specified passages are satisfied.

(b) Edit [Interrupt]

You can edit an interrupt vector and the priority order. The procedure for editing is the same as that for [\[Interrupt event setting\] tab \[Simulator\]](#) in the [Action Events dialog box](#) (see "(a) Specify the [Interrupt vector]" and "(b) Specify the [Priority order]").

2.17 Event Management

An event refers to a specific state of the microcontroller in debugging like "the address 0x1000 was fetched" or "data was written to the address 0x2000."

CubeSuite+ uses this event as an action trigger of the debug function to break at a given place, start or stop a trace operation, or start or stop a timer measurement.

This section describes how to manage those events.

The events are managed collectively on the event panel shown below.

Choose [Event] from the [View] menu.

The [Events panel](#) permits you to check detailed information on the currently set events in list form, as well as delete events, switch the state of settings (enabled or disabled), display detailed information, and change settings.

For details on how to read each area and about their functionality, see the section in which the [Events panel](#) is described.

Figure 2-142. Displaying the Set Events (Events Panel) [E1] [E20]

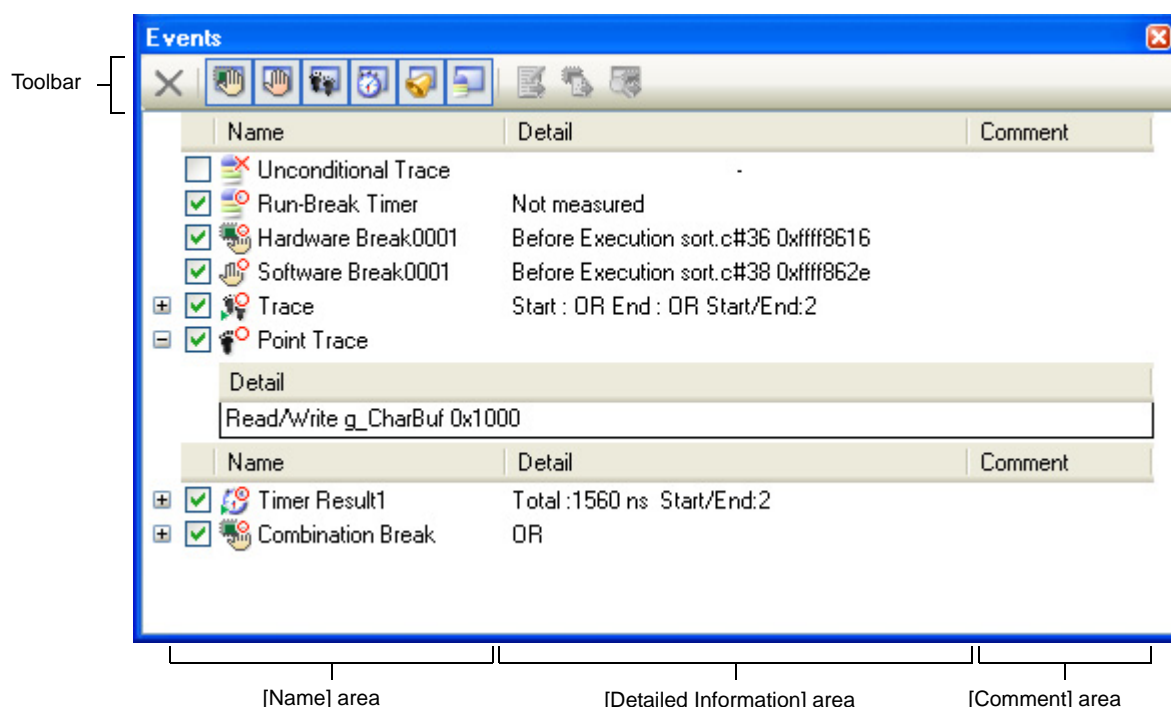
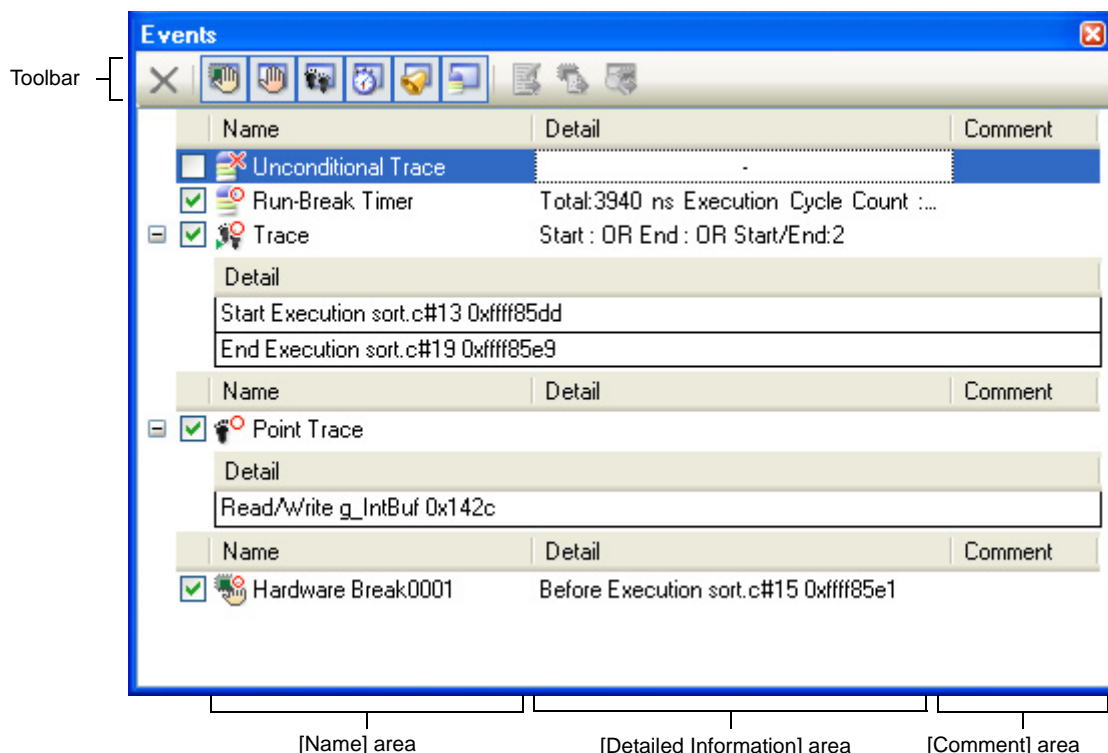


Figure 2-143. Displaying the Set Events (Events Panel) [Simulator]



2.17.1 Changing states of setting (Enabled/Disabled)

By checking or unchecking the checkbox of an event name concerned, it is possible to change the set state of that event. (When the set state of an event is changed, its [Event mark](#) is changed accordingly.)

There are following types for the set states of events.

Figure 2-144. Event Name Checkbox

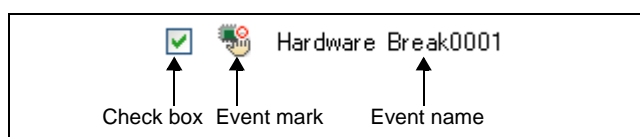


Table 2-21. Set States of Events

<input checked="" type="checkbox"/>	Enabled	When a specified condition is met, the enabled event occurs. The event can be disabled by unchecking its checkbox.
<input type="checkbox"/>	Disabled	Even when a specified condition is met, the disabled event does not occur. The event can be enabled by checking its checkbox.
<input type="checkbox"/>	Pending	A specified condition cannot be set in the program to be debugged. The checkbox of any event in this state cannot be manipulated.

Remarks 1. The Run-Break timer event cannot be disabled or left pending.







- The state of an event can also be changed by enabling or disabling it from the menu that is displayed by right-clicking its [Event mark](#) on the [Editor panel](#) or [Disassemble panel](#).
- The enabled/disabled settings of an unconditional trace event and other trace events are controlled exclusively of each other. Therefore, although the unconditional trace event, one of the built-in events,

is enabled by default, when either a trace start event or a trace end event is set, it is automatically disabled at the same time, in which case, the trace start and trace end events are enabled together as one instance of a trace event.

Conversely, when the set trace event is disabled, the unconditional trace event is automatically re-enabled.

2.17.2 Displaying only a specific type of event




By clicking one of the following toolbar buttons, it is possible to display only a specific type of event.

	Displays hardware break-related events.
 [E1] [E20]	Displays software break-related events.
	Displays trace-related events.
	Displays timer-related events.
	Displays action events (Printf event, interrupt event [Simulator]).
	Displays built-in events (unconditional trace event or Run-Break timer event).

2.17.3 Jump to an event address

By clicking one of the following buttons, it is possible to jump to the relevant panel corresponding to the address of a currently selected event.

However, these buttons are disabled when a trace event, timer measurement event, or a built-in event (unconditional trace event or Run-Break timer event) is selected.

	The Editor panel opens, with the caret on it moved to the source line corresponding to the address at which a selected event is set.
	The Disassemble panel opens, with the caret on it moved to the disassembled result corresponding to the address at which a selected event is set.
	The Memory panel opens, with the caret on it moved to the memory value corresponding to the address at which a selected event is set.

2.17.4 Editing detailed settings of events

This section describes how to edit detailed settings of various events. For information on editing of timer measurement events, see "(3) [Editing a timer measurement event](#)". For information on editing of action events (Printf events and interrupt events), see "[2.16 Set an Action into Programs](#)".

- (1) [Editing execution-related events](#)
- (2) [Editing access-related events](#)
- (3) [Editing combination conditions of events \[E1\] \[E20\]](#)

(1) Editing execution-related events

It is possible to edit the condition for the number of times an execution-related event has passed.

To edit execution-related events, use the [Detailed Settings of Execution Events dialog box](#) that is displayed by performing the following operation on the [Events panel](#).

Hardware break	Move the caret to a hardware break you want to edit and then select [Edit Condition ...] from the context menu.
----------------	---

Trace	Move the caret to an execution-related event in a trace you want to edit and then select [Edit Condition ...] from the context menu.
Combination break [E1] [E20]	Move the caret to an execution-related event in a combination break you want to edit and then select [Edit Condition ...] from the context menu.
Timer Result [E1] [E20]	Move the caret to an execution-related event in a timer result you want to edit and then select [Edit Condition ...] from the context menu.

Caution Address values of execution-related events cannot be changed in the [Detailed Settings of Execution Events dialog box](#). To change address values, delete the subject event and then set a new event.

(a) Editing the pass count (or the number of times passed)

Enter a numeric value in decimal notation for the [Pass Count] property in the [Pass Count] category and then click the [OK] button. The relevant event occurs when the event condition is satisfied as many times as the entered pass count.

The specifiable value for the pass count differs with each debug tool as follows:

- [E1] [E20]
1 to 256
- [Simulator]
1 to 16,383

Cautions 1. [E1] [E20]

The pass count can be specified by any value other than 1 for only one event among all events (including access-related events).

If a pass count of other than 1 is specified for two events, an error results.

2. [E1(Serial) [RX200 Series]] [E20(Serial) [RX200 Series]]

No values other than 1 can be specified for the pass count.

(2) Editing access-related events

The address condition and data condition for access-related events can be edited.

For details on how to edit the pass count, see "(a) [Editing the pass count \(or the number of times passed\)](#)".

To edit access-related events, use the [Detailed Settings of Access Events dialog box](#) that is displayed by performing the following operation on the [Events panel](#).

Trace	Move the caret to an access-related event in a trace you want to edit and then select [Edit Condition ...] from the context menu.
Point Trace	Move the caret to an access-related event in a point trace you want to edit and then select [Edit Condition ...] from the context menu.
Combination break [E1] [E20]	Move the caret to an access-related event in a combination break you want to edit and then select [Edit Condition ...] from the context menu.
Timer Result [E1] [E20]	Move the caret to an access-related event in a timer result you want to edit and then select [Edit Condition ...] from the context menu.

Caution Address values of access-related events cannot be changed in the [Detailed Settings of Access Events dialog box](#). To change address values, delete the subject event and then set a new event.

(a) Editing address conditions

In the [Compare Condition] property in the [Address Condition] category, specify a compare condition from the drop-down list.

Depending on the specified compare condition, the following address conditions can be set in combination with the address values displayed in the [Address] property in the [Address] category.

<1> When [No conditions] is specified

The address value is made the condition. When an access to the address value occurs, the condition holds true.

<2> When [Address area] is specified [E1] [E20]

The address value is made the start address. Enter an end address value in the [End Address] property that is added in the [Address Condition] category. When access to any place in the range of start and end addresses set occurs, the condition holds true.

Also, in the [Area condition] property that is added the same way, it is possible to set "outside" of the address range for the address condition.

Cautions 1. [RX200 Series]

Compare conditions based on an address area are not supported

2. [RX600 Series]

Compare conditions based on an address area can set only once.

3. [Simulator]

Address conditions cannot be specified.

<3> When [Compare with address mask] is specified [E1] [E20]

A mask value can be set for the address value. Enter an address mask value in hexadecimal notation in the [Address mask] property that is added in the [Address Condition] category. When an access to the address that matches the masked address occurs, the condition holds true.

Also, in the [Compare] property that is added the same way, it is possible to set "Any other value (!=)" for the address compare condition.

Caution When specifying a mask value, be sure that the address value used as condition is masked bitwise, wherein the mask value "0" is "Don't Care."

Example) To set an address condition of 0x1000 to 0x1FFF

Address value: 0x1000

Mask value: 0xF000

(b) Editing data conditions

<1> [Access type]

Specify one of the following items for the access type

Read	When a read access occurs, the condition holds true.
Write	When a write access occurs, the condition holds true.
Read/Write	When a read or write access occurs, the condition holds true.

<2> [Access size]

Specify one of the following items for the access size.

No conditions	When an access in any size occurs, the condition holds true.
Byte	When an access in byte size occurs, the condition holds true.

Word	When an access in word size occurs, the condition holds true.
Long word	When an access in long word size occurs, the condition holds true.

<3> [Compare condition]

Specify a data comparison condition.

The comparison condition differs with each debug tool as follows:

[E1] [E20]	Specified value (==)	When data compare conditions match, the condition holds true.
	Any other value (!=)	When data compare conditions do not match, the condition holds true.
[Simulator]	Equal (==)	When data and specified value match, the condition holds true.
	Not equal (!=)	When data and specified value do not match, the condition holds true.
	Inverse sign	When the sign is inverted between the previously accessed data and the data accessed this time, the condition holds true. Note
	Difference	When the difference between the previously accessed data and the data accessed this time exceeds a specified value, the condition holds true. Note
	Greater than (>)	When data is greater than a specified value, the condition holds true.
	Less than (<)	When data is smaller than a specified value, the condition holds true.
	Greater than or equal to (>=)	When data is equal to or greater than a specified value, the condition holds true.
	Less than or equal to (<=)	When data is equal to or smaller than a specified value, the condition holds true.
	Inside the range (<=Values<=)	When data is within the range of specified values, the condition holds true. ([Compare data 1] <= data <= [Compare data 2])
	Outside the range !(<=Values<=)	When data is outside the range of specified values, the condition holds true. (data < [Compare data 1] [Compare data 2] < data)
	No conditions	Comparison data is not specified.

Note For [Inverse sign] and [Difference], since comparison is made with the previous data, the condition never holds true after a reset and in the first determination of whether condition is true.

Remarks 1. [E1] [E20]

This property is displayed only when you've specified [Yes] in the [\[Specify the data mask\]](#) property.

2. [Simulator]

This property is not displayed only when you've specified [No conditions] in the [\[Access size\]](#) property.

<4> [Compare data] [E1] [E20]

Specify a data value with a hexadecimal number.

<5> [Compare data1] [Simulator]

Specify a data value in hexadecimal.

This property is displayed when you've specified one of the following items in the [\[Compare condition\]](#) property.

[Equal (==)], [Not equal (!=)], [Greater than (>)], [Less than (<)], [Greater than or equal to (>=)], [Less than or equal to (<=)], [Inside the range (<=Values<=)], [Outside the range !(=Values<=)]

<6> [Compare data2] [Simulator]

Specify a data value with a hexadecimal number.

This property is displayed only when you've specified [Inside the range (<=Values<=)] or [Outside the range !(=Values<=)] in the [\[Compare condition\]](#) property.

<7> [Specify the data mask]

When you entered a comparison data value, this property is displayed, allowing you to specify whether or not a mask value for the data value be specified.

Remark [Simulator]

This property is displayed when you've specified one of the following items in the [\[Compare condition\]](#) property.

[Equal (==)], [Not equal (!=)], [Greater than (>)], [Less than (<)], [Greater than or equal to (>=)], [Less than or equal to (<=)], [Inside the range (<=Values<=)], [Outside the range !(=Values<=)]

<8> [Mask value]

This property is displayed when you've specified [Yes] in the [\[Specify the data mask\]](#) property. Enter a data mask value in hexadecimal, so that when a data access that matches the masked data and comparison condition occurs, the condition holds true.

<9> [Difference] [Simulator]

Specify comparison data with a hexadecimal number.

This property is displayed when you've specified [Difference] in the [\[Compare condition\]](#) property.

<10> [Sign] [Simulator]

Specify whether or not the data to compare has a sign.

This property is displayed when you've specified one of the following items in the [\[Compare condition\]](#) property.

[Difference], [Greater than (>)], [Less than (<)], [Greater than or equal to (>=)], [Less than or equal to (<=)], [Inside the range (<=Values<=)], [Outside the range !(=Values<=)]

(3) Editing combination conditions of events [E1] [E20]

Edit the combination condition for a combination break comprised of multiple events set or for a trace. The combination condition for a trace can be set by specifying a condition for start and for end, respectively, by switching tabs.

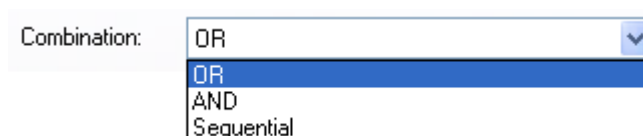
To edit the combination condition of events, use the Set [Combination Condition dialog box \[E1\]\[E20\]](#) that is displayed by the following operation on the [Events panel](#).

Combination break	Move the caret to combination break and then select [Edit Condition ...] from the context menu.
Trace	Move the caret to a trace and then select [Edit Condition ...] from the context menu.

(a) Editing the combination condition

Select the combination condition between [OR], [AND] or [Sequential] from the pulldown menu. For the explanation of operation per combination condition, see "[2.10.5 Set multiple break events in combination \(Combination break\) \[E1\] \[E20\]](#)", and "[\(2\) Combining multiple events \[E1\] \[E20\]](#)".

Figure 2-145. [Combination] item

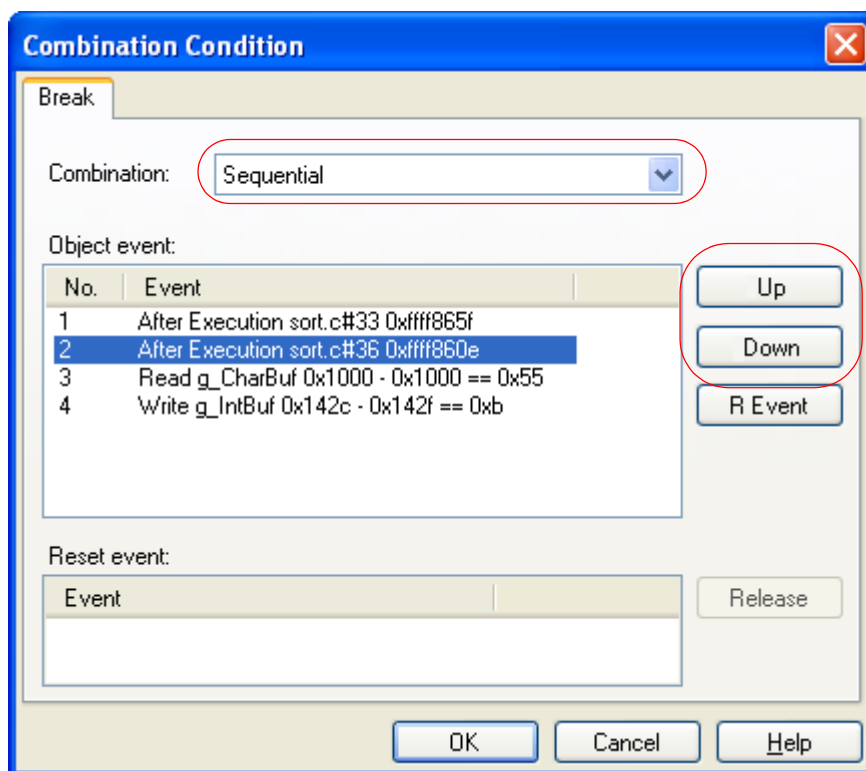


- Cautions**
1. For trace events, the combination condition available for a trace end event is only [OR].
 2. [AND] and [SEQUENTIAL] can be used as a combination condition by the combination break or Trace start event on a mutually exclusive basis. So when [AND] and [SEQUENTIAL] are selected as a combination condition for either of the events, only [OR] can be selected for the other.

(b) Editing the subject event

Change the order of events in the subject event list using the [Up] or [Down] button. These buttons are enabled only when you've selected Sequential for the combination condition and moved the caret to an event in the subject event list.

Figure 2-146. Changing the Order of Events in the Subject Event List (Sequential)

**(c) Editing the reset event**

To register a reset event, click the [R Event] button. This button is enabled only when you've selected Sequential for the combination condition and moved the caret to an event in the subject event list.

Also, if you want to clear registration of a reset event, click the [Release] button. This button is enabled when you move the caret to an event in the reset event list.

Cautions 1. Only one reset event is specifiable.

If you press the [R Event] button while reset events are registered, the event selected in the subject event list and the registered reset events are exchanged for each other.

- If you change the combination condition from [Sequential] to [OR] or [AND] while reset events are registered, events in the reset event list are excluded from the subject event. If you want some of them the subject event, clear registration of the reset event before you change the combination condition.
- An event for which the pass count has been specified cannot be registered as the reset event. The pass count cannot be specified for an event that has been registered as the reset event.
- [RX600 Series]
An event for which the address area has been specified cannot be registered as the reset event. The address area cannot be specified for an event that has been registered as the reset event.

Figure 2-147. Registering a Reset Event (Sequential)

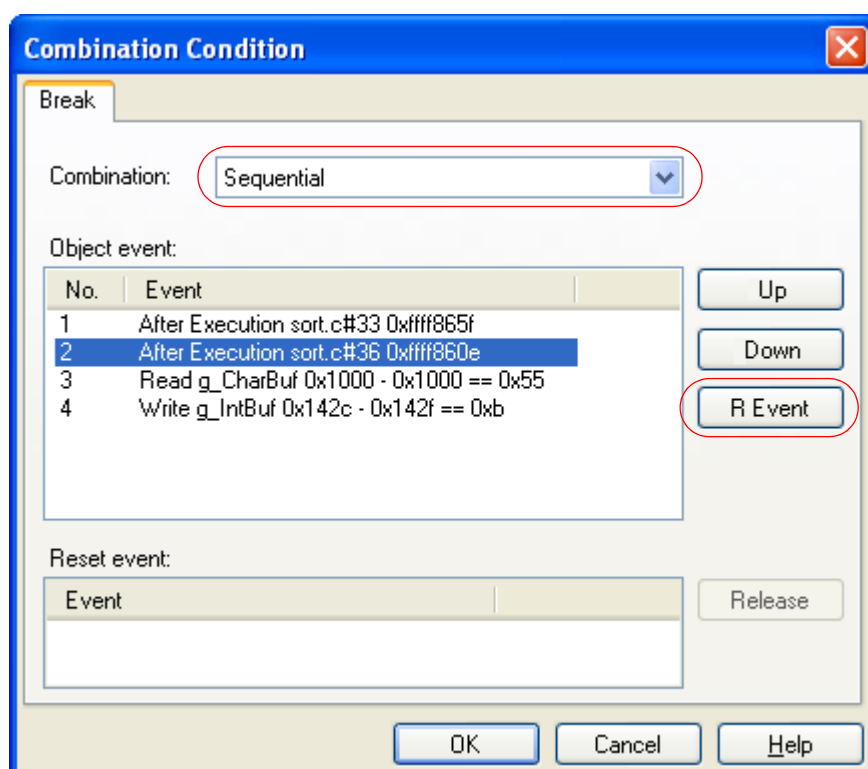
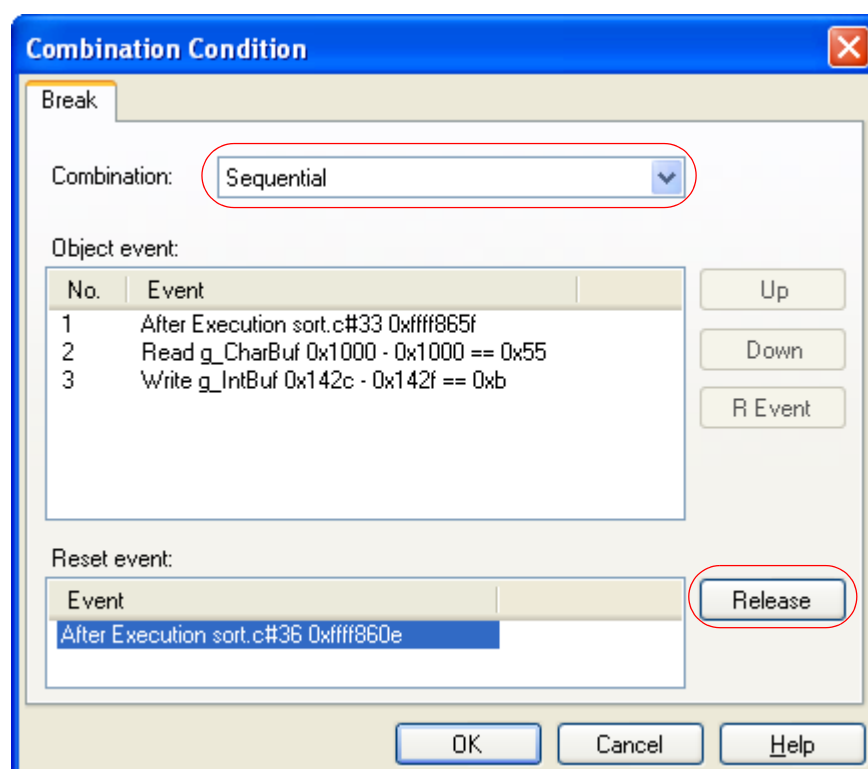



Figure 2-148. Releasing Registration of a Reset Event (Sequential)




2.17.5 Deleting events

To delete any event you've set, select the subject event and then click the toolbar button .

However, you cannot delete the unconditional trace event and Run-Break timer event, which are the built-in events of the debug tool.

Remarks 1. [E1] [E20]

- For the timer event, you can delete an event by clicking its event mark that is displayed on the [Editor panel](#) or [Disassemble panel](#).
- For the software/hardware break and the trace event consisting of execution-related events, you can delete an event by clicking their event mark that is displayed on the [Editor panel](#) or [Disassemble panel](#).
 - To delete all of the events you've set at a time, select [Select All] on the context menu and then click the  button (not including the built-in events).

2.17.6 Entering comments for events

For any event you've set, you can enter a comment freely as you wish.

To enter a comment, select an event for which you want to enter a comment, click in the [\[Comment\] area](#), and then enter any text directly from the keyboard. (Pressing the [Esc] key cancels the edit mode.)

After editing a comment, press the [Enter] key or move the focus to other than the editing area to complete your editing.

A comment can be entered in up to 256 characters, which is saved as the setting of the current user.

2.17.7 Points to note regarding event setting

Here, the points to note when you make settings of various events are described.

- [\(1\) Limitation on the number of valid events](#)
- [\(2\) Types of events that can be set or removed during execution](#)
- [\(3\) Other precautions](#)

(1) Limitation on the number of valid events

The number of events that can be set as valid (or [Enabled](#)) events at the same time are subject to the following limitations.

Therefore, if this limit is exceeded when you set one or more new valid events, some of the events that are already set need to be [Disabled](#) before you can set a new one.

Table 2-22. Limitation on the Number of Valid Events [RX600 Series]

Type of event	Debug tool	
	E1(Serial)/E1(JTAG) E20(Serial)/E20(JTAG)	Simulator
Software break	256	0
Hardware break (execution-related: before Go)	8 ^{Note1}	1024
Break event (execution-related: after Go)	8 ^{Note1}	0
Break event (access-related: Read, Write, Read/Write)	4 ^{Note1 Note3}	1024
Combination break	8+4 ^{Note1 Note2 Note5}	0
Trace (trace start, trace end)	8+4 ^{Note1 Note2 Note6}	256
Point trace	4 ^{Note1 Note2}	256
Timer result (timer start, timer end)	8+4 ^{Note1 Note2 Note6}	0

Type of event	Debug tool	
	E1(Serial)/E1(JTAG) E20(Serial)/E20(JTAG)	Simulator
Action (Printf event)	Same as for the number of software breaks	Same as for the number of hardware breaks
Action (Interrupt event)	0	256

Table 2-23. Maximum Number of Enabled Events [RX200 Series]











Type of event	Debug tool	
	E1(Serial) E20(Serial)	Simulator
Software break	256	0
Hardware break (execution-related: before Go)	⁴ Note4	1024
Break event (execution-related: after Go)	⁴ Note4	0
Break event (access-related: Read, Write, Read/Write)	² Note4	1024
Combination break	⁴⁺² Note4	0
Trace (trace start, trace end)	⁴⁺² Note4 Note5 Note6	256
Point trace	⁴ Note4	256
Timer result (timer start, timer end)	⁴⁺² Note4 Note6	0
Action (Printf event)	Same as for the number of software breaks	Same as for the number of hardware breaks
Action (Interrupt event)	0	256

- Notes**
- There are 8 events for execution address and 4 events for data access, which are shared in each function.
 - For an event combination, up to 8 events can be set.
 - [E20(JTAG)]**
The trace function and the realtime RAM monitor function (RRM function) in part are usable exclusively of each other. Therefore, while the RRM function is in use, the point trace cannot be used. Also, the access event is usable only when the RRM block has free space.
 - There are 4 events for execution address and 2 events for data access, which are shared in each function.
 - If the combination condition for the combination break is Sequential, the maximum number of events that can be set differs with each microcontroller used, as shown below.
 - **[RX600 Series]**
7 events + reset event (R event)
 - **[RX200 Series]**
3 events + reset event (R event)
 - If the maximum number of execution-related events have already been used as trace start/trace stop events or timer start/timer stop events, [Go to Here] in the context menu of the [Editor panel](#) or [Disassemble panel](#) is not usable. Also, it is not possible to run the program to the address of a specified symbol after the CPU has been reset.

(2) Types of events that can be set or removed during execution


Types of events that can be set or removed during execution of the program or during execution of the tracer/timer are as follows:

Table 2-24. Types of Events that Can Be Set or Removed during Execution

Type of event	Debug tool	
	E1(Serial)/E1(JTAG) E20(Serial)/E20(JTAG)	Simulator
Software break	 Note1 Note2	—
Hardware break (execution-related: before Go)	 Note2	
Break event (execution-related: after Go)	 Note2	—
Break event (access-related: Read, Write, Read/Write)	 Note2	
Trace (trace start, trace end)	×	
Point trace	×	
Timer result (timer start, timer end)	×	—
Action (Printf event)	 Note1 Note2	×
Action (Interrupt event)	—	

 : Possible

 : Possible if program execution is momentarily halted

 : Impossible during tracer/timer execution

— : Not supported

×

 : Impossible

Notes 1. Internal RAM area only

2. Events cannot be set while the emulator is in a power-down mode.

(3) Other precautions

- No events can be set to local variables.

- No events occur during single-stepping (including Return-out), as well as during program execution actuated by selecting [Go to Here] from the context menu.

- If the program to be debugged is downloaded again and the position at which some existing event is set happens, as a consequence of it, to be in the middle of an instruction, the relevant event needs to be reset. Here is the method for doing it.

- When debug information is available:

The event set position always moves to the beginning of source text lines.

- When no debug information is available:

The position depends on how the [Automatic change method of event setting position] property in the [Download] category on the Property panel's [Download File Settings] tab is set.

2.18 Setting Up the Hook Process

This section describes how to set up hooks in the debug tool by using the hook transaction function.

By setting up hook transaction, it is possible to automatically change the I/O register and CPU register values before or after downloading a load module or after resetting the CPU.

You can configure hook transaction in the [\[Hook Transaction Settings\]](#) category of the [\[Hook Transaction Settings\]](#) tab on the [Property panel](#).

Remark You can increase the download speed by configuring the I/O register in the [\[Before download\]](#) property. Equally, you can facilitate downloading to external RAM by the setting in this category.

Figure 2-149. [Hook Transaction Settings] Category

Hook Transaction Settings	
Before download	Before download[0]
After download	After download[0]
After CPU reset under breaking	After CPU reset under breaking[0]
Before running	Before running[0]
After breaking	After breaking[0]

Table 2-25. Properties of the [Hook Transaction Settings] Category

Properties	Timing
Before download	Performs a specified process immediately before load module files are downloaded.
After download	Performs a specified process immediately after load module files are downloaded.
After CPU reset under breaking	Performs a specified process immediately after the CPU is reset.
Before running	Performs a specified process immediately before program execution starts.
After breaking	Performs a specified process immediately after program execution breaks.

Each property in the [\[Hook Transaction Settings\]](#) category shows the timing with which a hook transaction is performed. The numeral shown in bracket [] for each property value indicates the number of currently specified transactions. (No hook transactions are set by default.)

Follow the procedure below to perform hook transaction on the desired property.

To specify the transaction, select the desired property and click the [...] button that appears on the right end of the column. This will open the [Text Edit dialog box](#) in which you can specify the transaction.

Figure 2-150. Text Edit Dialog Box Opened

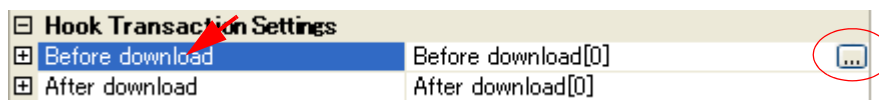
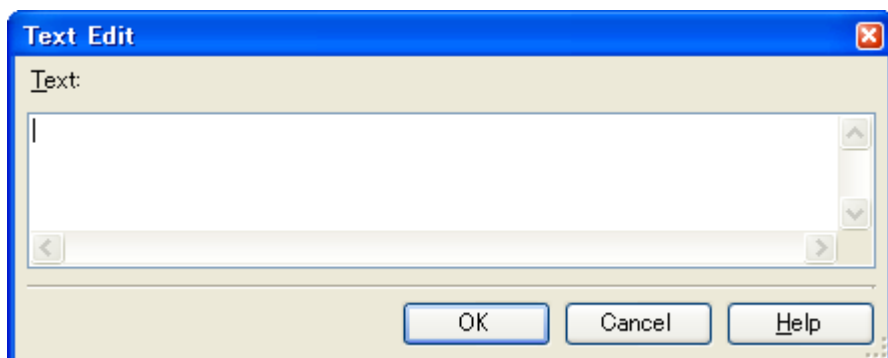


Figure 2-151. Configuring Hook Transaction (Text Edit Dialog Box)



In this dialog box, enter the desired transaction directly from the keyboard.
You can use one of the following formats for specifying the desired transaction.

[Format: 1]

Overwrites the *I/O register* contents with numeric values.

```
I/O register name Numeric value
```

[Format: 2]

Overwrites the *CPU register* contents with numeric values.

```
CPU register name Numeric value
```

[Format: 3]

Executes a script file designated by *Python script path* (absolute path or relative path using project folder as a base).

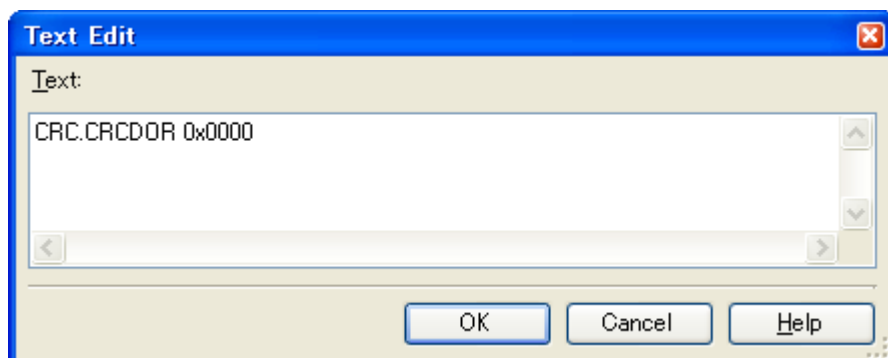
```
source Python script path
```

- Remarks**
1. When specifying the transaction, add "#" at the top of the line to comment it out.
 2. The space can be substituted by a tab character.

You can enter up to 64 characters for one transaction and can specify up to 128 transactions for each property. (One line in the [Text] area in the Text Edit dialog box corresponds to one transaction.)

When finished with specifying the transaction, click [OK] to reflect it in the Property panel.

Figure 2-152. Example of Hook Transaction Setting

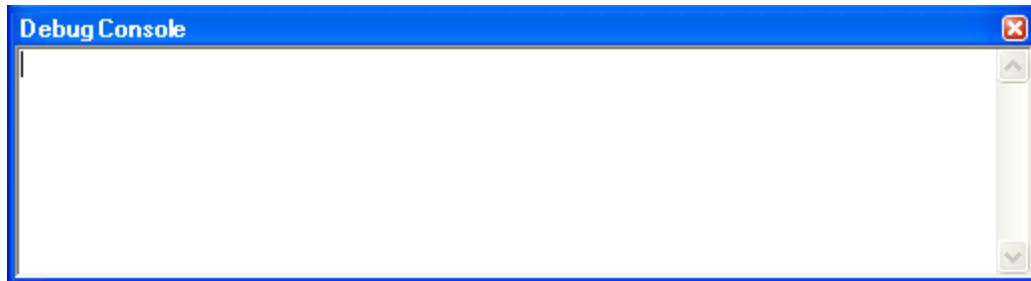


2.19 Using the Debug Console

This section describes how to use the debug console for standard input/output.

To perform some operation using a program's internal standard library functions, such as the `scanf` function for reading the data entered from the keyboard or the `printf` function for outputting data, use the [Debug Console panel](#) shown below.

Figure 2-153. Debug Console Panel



For details about each functionality of the debug console, see the section in which the [Debug Console panel](#) is described.

Before standard input/output based on the [Debug Console panel](#) can be used in a C/C++ program, there must be a low-level interface routine in the program.

CubeSuite+ comes with sample project "RX610_Tutorial_DebugConsole" for the E1/E20, which has the following files for debug console functions.

- Low-level interface routine files (C language part) (`lowsrc.c` and `lowsrc.h`)
 - Low-level interface routines: `open`, `close`, `read`, `write`, and `lseek`
Called by standard input/output.
 - Initialize input/output libraries: `_INIT_IOLIB`
 - Close all open functions: `_CLOSEALL`
Each called by the internal initialization routine "PowerON_Reset_PC" of the initialization routine file `resetprg.c`.
- Low-level interface routine file (assembly language part) (`lowlvl.src`)
 - Function to output 1 character: `_charput`
 - Function to input 1 character: `_charget`
Called by low-level interface routines "write" and "read," respectively.
- Source file including the main function (`DebugConsole_Sample.c`)
 - Calls the standard library functions `scanf` and `printf` in the main function.

Caution [E1] [E20]

To use the debug console, do not switch the system clock in the `charput` and `charget` functions of the program. Switching of the system clock may adversely affect the communication between the emulator and microcontroller, hampering normal transmission or reception of data.

Remarks 1. For details about the low-level interface routines, see Chapter 7, "Startup," in the separate edition, "CubeSuite+ RX Coding."

2. [Simulator]

For details about the simulator's input/output functions, see "[APPENDIX B INPUT/OUTPUT FUNCTIONS](#)."

To use sample projects with debug console functions, follow the procedure described below.

(1) Loading a sample project

From [Open Sample Project] on the Start panel of CubeSuite+, load sample project "RX610_Tutorial_DebugConsole" for the E1/E20.

Remark For details about the "Start panel," see the separate edition, "CubeSuite+ Start."

(2) Editing the low-level interface routine file (assembly language part) [Simulator]

When the simulator is to be used, the low-level interface routine file "lowlvl.src" (assembly language part) must be edited so that it will be specific to the simulator. Replace the contents of "lowlvl.src" with the following sample code for the simulator.

```

        .GLB    _charput
        .GLB    _charget

SIM_IO   .EQU 0h

        .SECTION    P, CODE

; -----
;  _charput:
; -----
_charput:
        MOV.L    #IO_BUF, R2
        MOV.B    R1, [R2]
        MOV.L    #1220000h, R1
        MOV.L    #PARM, R3
        MOV.L    R2, [R3]
        MOV.L    R3, R2
        MOV.L    #SIM_IO, R3
        JSR      R3
        RTS

; -----
;  _charget:
; -----
_charget:
        MOV.L    #1210000h, R1
        MOV.L    #IO_BUF, R2
        MOV.L    #PARM, R3
        MOV.L    R2, [R3]
        MOV.L    R3, R2
        MOV.L    #SIM_IO, R3
        JSR      R3
        MOV.L    #IO_BUF, R2
        MOVU.B    [R2], R1
        RTS

; -----
;  I/O Buffer
; -----

        .SECTION    B, DATA, ALIGN=4
PARM:    .BLKL      1
        .SECTION    B_1, DATA
IO_BUF:  .BLKB      1
        .END

```

(3) Selecting the debug tool to be used

To select or switch the debug tool, use the context menu shown by right clicking on the [*Microcontroller type Debug tool name (Debug Tool)*] node on the [Project Tree](#) panel.

Remark For the details of debug tool selection, see "[2.3.1 Select the debug tool to use](#)".

(4) Changing settings in the Property panel [Simulator]

When the simulator is to be used, the following settings must be made in the [\[Stream I/O\] \[Simulator\]](#) category on the [\[Debug Tool Settings\]](#) tab of the [Property](#) panel.

Figure 2-154. [\[Stream I/O\]](#) Category

**(a) [Use stream I/O function]**

If you intend to perform standard input/output or file input/output, select [Yes] (default selection: [No]).

(b) [Stream I/O address]

If you have replaced the low-level interface routines (assembly-language part) with the sample code for the simulator, enter "0" (default).

(5) Executing a download

Choose [Build & Download] from the [Debug] menu on the Main window. Then, when connected with the debug tool, perform a build and a download (see ["2.5.1 Execute downloading"](#)).

(6) Opening the Debug Console panel

Open the [Debug Console](#) panel.

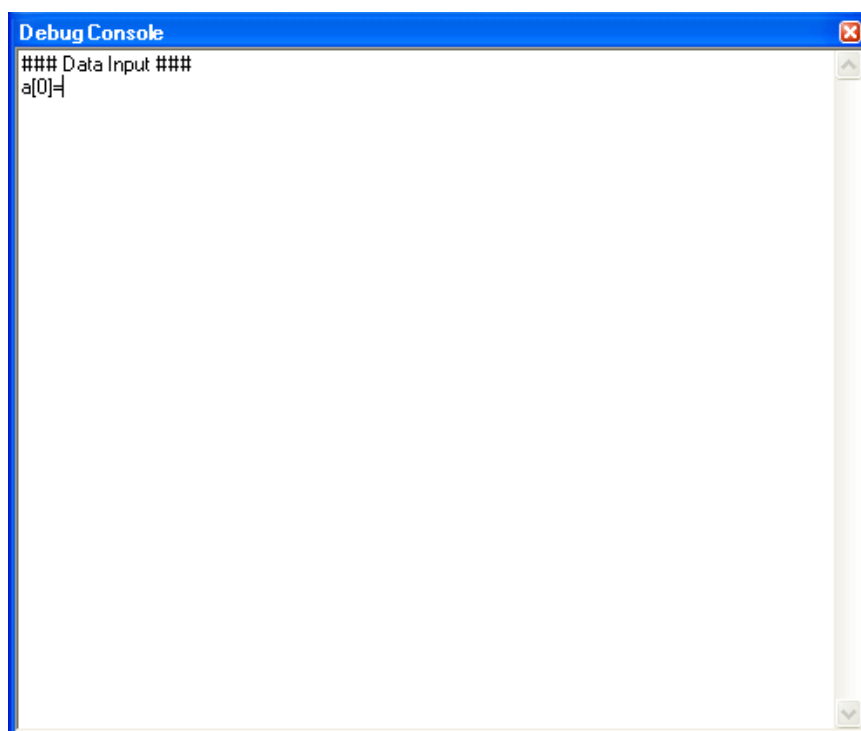
(7) Executing the program

Execute the program (see ["2.9 Execute Programs"](#)).

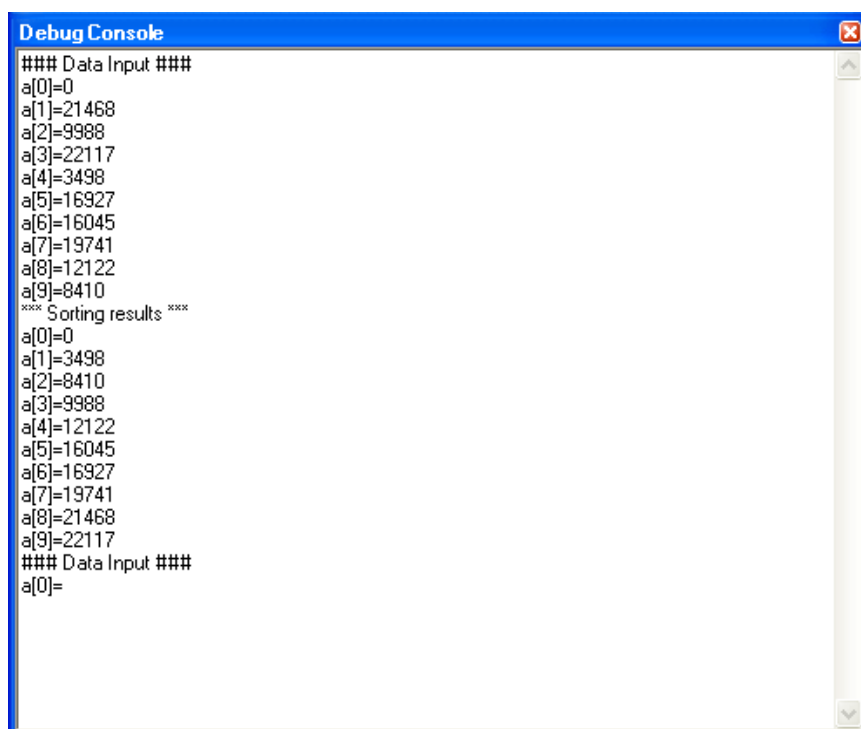
(8) Performing data input/output on debug console panel

By executing the program, perform data input/output on the [Debug Console](#) panel.

This program reads 10 entries of data that have been entered from the keyboard onto the panel by the scanf function and then sorts the input data in ascending order. It then outputs data onto the panel using the printf function.



- The element names of the array "a" in which data is stored are output onto the panel by the printf function in the main function.
- When data is entered from the keyboard onto the panel, the data is read in by the scanf function in the main function.



- The sorted data is output onto the panel by the printf function in the main function.

Remark If you disable the local echo back function in [Echo Back] selected from the [Debug Console panel's](#) context menu, the character strings you have entered will not be displayed (local echo back will not be performed.)

2.20 About Input Value

This section describes consideration to take when inputting values in each panel and dialog box.

2.20.1 Input rule

Following is the rules for input to each panel and dialog box.

(1) Character set

Character sets that are allowed to input are as follows:

Table 2-26. List of Character Set

Character Set	Outline
ASCII	1- byte alphabets, numbers, symbols.
Shift-JIS	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji and 1-byte Katakana.
EUC-JP	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji and 1-byte Katakana.
UTF-8	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji (include Chinese characters) and 1-byte Katakana.
UTF-16 (Unicode)	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji (include Chinese characters) and 1-byte Katakana.

(2) Escape sequence

Escape sequences that are allowed to input are as follows:

Table 2-27. Escape Sequence List

Escape sequence	Value	Description
\0	0x00	null character
\a	0x07	Alert
\b	0x08	Backspace
\t	0x09	Horizontal tab
\n	0x0A	New line
\v	0x0B	Vertical tab
\f	0x0C	Form feed
\r	0x0D	Carriage return
\"	0x22	Double-quotation mark
\'	0x27	Single-quotation mark
\?	0x3F	Question mark handled as a question mark if ? is entered.
\\	0x5C	Backslash

(3) Number

Notations allowed when entering numbers are as follows:

Table 2-28. Notation List

Notation	Outline
Binary number	Start with 0b and continues with the numbers from 0 to 1. (Case insensitive for alphabets)
Octal number	Start with 0 and continues with the numbers from 0 to 7.
Decimal number	Start without 0 and continues with the numbers from 0 to 9.
Hexadecimal number	Start with 0x and continues with the numbers from 0 to 9 and alphabets a to f. (Case insensitive for alphabets) In the input area with the HEX mark, prefix 0x is not needed.

(4) Expression and operator

Expression represents constants, CPU register name, I/O register name and symbols and those connected with operators.

An expression comes in two types: an address expression and a watch expression. The expression that requires the address of a symbol is referred to as an address expression, and the one that requires the value of a symbol is referred to as a watch expression.

(a) An address expression and operators

With an address expression, the address of a symbol is used to perform operations. Only when a CPU register name is written, the value of the symbol is used to perform operations.

The basic input formats of address expressions are as follows:

Table 2-29. Basic Input Format of Address Expressions

Expression	Description
Name of a C/C++ language variable ^{Note 1 No 2}	Address of a C/C++ language variable
<i>Expression</i> [<i>Expression</i>] ^{Note 3}	Address of an array
<i>Expression</i> .Member name ^{Note 4}	Address of a structure/union/class member
<i>Expression</i> ->Member name ^{Note 4}	Address of a structure/union/class member that is pointed to
Watch expression.*Cast expression	Address of the pointer to a member variable
Watch expression->*Cast expression	Address of the pointer to a member variable
Name of a CPU register	Value of the CPU register
Name of an I/O register	Address of the I/O register
Label name ^{Note 5} , EQU symbol name ^{Note 5} , and [immediate value]	Address of a label, a value of an EQU symbol, and an immediate address
Integer constant	Address

Notes 1. C89, C99, or C++ language variable

2. If the register is assigned the value of a C variable, an error results.

3. The expression that is input as an index to an array is parsed as a watch expression.

4. When using a member variable of a base class, specify the scope before the member name (e.g. variable.BaseClass::member).

5. If the label name or EQU symbol name includes a "\$," be sure to enclose the name in "{ }."
(Example: {\$Label})

When you specify the CPU register name "I", add ":REG" (e.g. I:REG) to distinguish it from the keyword "I" that indicates an imaginary number.

From "Table 2-29. Basic Input Format of Address Expressions", the following address expressions with operator can be constructed.

Table 2-30. Construction of Address Expressions with Operators

Expression	Description
(Expression)	Specifies the order in which operations are performed
-Expression	Inverts symbol
!Expression	Logical negation
~Expression	Bit inversion
Expression * Expression ^{Note}	Multiplication
Expression / Expression ^{Note}	Division
Expression % Expression ^{Note}	Remainder calculation
Expression + Expression ^{Note}	Addition
Expression - Expression ^{Note}	Subtraction
Expression & Expression ^{Note}	Logical multiplication by bits
Expression ^ Expression ^{Note}	Exclusive disjunction by bits
Expression Expression ^{Note}	Logical sum by bits

Note Variables and functions can be combined by an operator only with variables, functions and integer constants. (Example: C variable name + IOR name)

(b) Watch expression and operator

With watch expression, the value of a symbol is used to perform operations. Only when the value does not exist, the address of the symbol is used to perform operations. (Example: main() + 1)

The basic input formats of watch expressions are as follows:

Table 2-31. Basic Input Format of Watch Expressions

Expression	Description
Name of a C/C++ language variable ^{Note 1}	Value of a C/C++ language variable
Expression[Expression]	Element of an array
Expression.Member name ^{Note 2}	Value of a structure/union/class member
Expression->Member name ^{Note 2}	Value of a structure/union/class member that is pointed to
Watch expression.*Cast expression	Value of the pointer to a member variable
Watch expression->*Cast expression	Value of the pointer to a member variable
*Expression	Value of pointer variable
&Expression	Location address
(Type name) Watch expression	Value cast into a specified type

Expression	Description
Name of a CPU register	Value of the CPU register
Name of an I/O register	Value of the I/O register
Label name ^{Note 3} , EQU symbol name ^{Note 3} , and [immediate value]	Values of a label and an EQU symbol, and an immediate address
Integer constant	Integer constant value
Floating constant	Floating point constant value
Character constant	Character constant value

Notes 1. C89, C99, or C++ language variable

2. When using a member variable of a base class, specify the scope before the member name (e.g. variable.BaseClass::member).

3. If the label name or EQU symbol name includes a "\$," be sure to enclose the name in "{ }." (Example: {\$Label})

Any imaginary number must be multiplied by an uppercase "I" (e.g. 1.0 + 2.0*I). When you specify the CPU register name "I", add ":REG" (e.g. I:REG) to distinguish it from the keyword "I" that indicates an imaginary number.

From "Table 2-31. Basic Input Format of Watch Expressions", the following watch expressions with operator can be constructed. For the operators listed in the table below, the expression is parsed according to C language specifications.

Table 2-32. Construction of Watch Expressions with Operators

Expression	Description
(Expression)	Specifies the order in which operations are performed
-Expression	Inverts symbol
!Expression	Logical negation
~Expression	Bit inversion
Expression * Expression ^{Note}	Multiplication
Expression / Expression ^{Note}	Division
Expression % Expression ^{Note}	Remainder calculation
Expression + Expression ^{Note}	Addition
Expression - Expression ^{Note}	Subtraction
Expression & Expression ^{Note}	Logical multiplication by bits
Expression ^ Expression ^{Note}	Exclusive disjunction by bits
Expression Expression ^{Note}	Logical sum by bits

Note Variables and functions can be combined by an operator only with variables, functions and integer constants. (Example: C variable name + IOR name)

2.20.2 Symbol name completion function

This function helps users input data by selecting one of the listed symbol names that exist in the program, when specifying an address expression and so on.

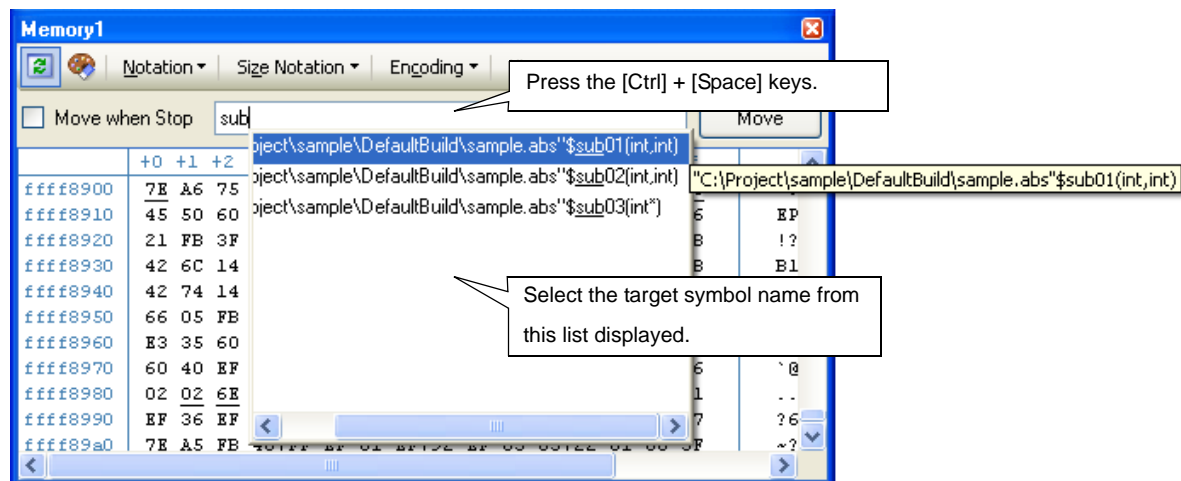
The list of symbol names appears by pressing the [Ctrl] + [Space] keys when a part of the target symbol name is being input in the text box that supports this function. In this list, double-click the target symbol name (or press the [Space]/[Enter] key after selecting it by using the [Up]/[Down] key) to complement the symbol name currently being input.

At this time, if a key other than the [Space]/[Enter] key is pressed or the focus moves to outside the panel/dialog box currently being operated, then the list of symbol names will disappear (the symbol name completion will not be performed).

- Cautions 1.** If there are no character strings in the text box or there are no candidates of the symbol, then the list of symbol names will not appear.
- 2.** Optimization by the compiler may convert the type of a function parameter to be different from that defined in the program (e.g. `func (class Foo obj) -> func (class Foo *obj);`). If this is the case, the symbol name completion provides the function names after optimization.


Remark See the explanation of the corresponding panel/dialog box as to whether this function can be used or not when inputting a symbol name.

Figure 2-155. Symbol Name Completion Function



2.20.3 Icons for invalid input

In some of the dialogs provided by CubeSuite+, the  icon will appear at a point where incorrect characters are entered as a warning sign.

Remark Placing the cursor over the  icon will pop up the information that indicates the characters to be entered.

APPENDIX A WINDOW REFERENCE

Appendix A provides detailed explanations of windows/panels/dialog boxes used for debugging with CubeSuite+.

A.1 Description

Windows/panels/dialog boxes for debugging are listed below.

Table A-1. Window/Panel/Dialog Box List

Window/Panel/Dialog Box Name	Description
Main window	Controls the program execution. Various windows, panels and dialogs can be opened from this window.
Project Tree panel	Selects the debug tool to use.
Property panel	Displays detailed information on the debug tool currently selected in the Project Tree panel , and enables the settings of the tool to be changed.
Editor panel	Enables text files to be viewed and edited, and is used to execute source level debug.
Memory panel	Displays and modifies memory values.
Disassemble panel	Displays the results of memory value disassemble and is used to execute line assemble and instruction level debug.
CPU Register panel	Displays the contents of CPU registers, and modifies register values.
IOR panel	Displays and modifies I/O register values.
Local Variables panel	Displays and modifies local variables.
Watch panel	Displays and modifies registered watch-expression values.
Call Stack panel	Displays call stack information on function calls.
Trace panel	Displays trace data acquired from the debug tool.
Events panel	Displays detailed information on set events, switches the events between enabled and disabled, or deletes them.
Output panel	Displays messages output from the build tool/debug tool/plugin-ins, or the results of batch searches carried out using the Find and Replace dialog box.
Debug Console panel	Performs standard input/output.
Memory Mapping dialog box [Simulator]	Sets the memory mapping.
Download Files dialog box	Selects files to be downloaded and sets the download conditions.
Text Edit dialog box	Inputs and modifies character strings.
Action Events dialog box	Sets action events.
Encoding dialog box	Selects a file-encoding.
Save Settings dialog box	Specifies the encoding and the new line code of the file being edited.
Memory Initialize dialog box	Initializes memory.
Memory Search dialog box	Searches memory.
Print Address Range Settings dialog box	Sets the address range to print the contents of the Disassemble panel .
Trace Search dialog box	Searches trace data.

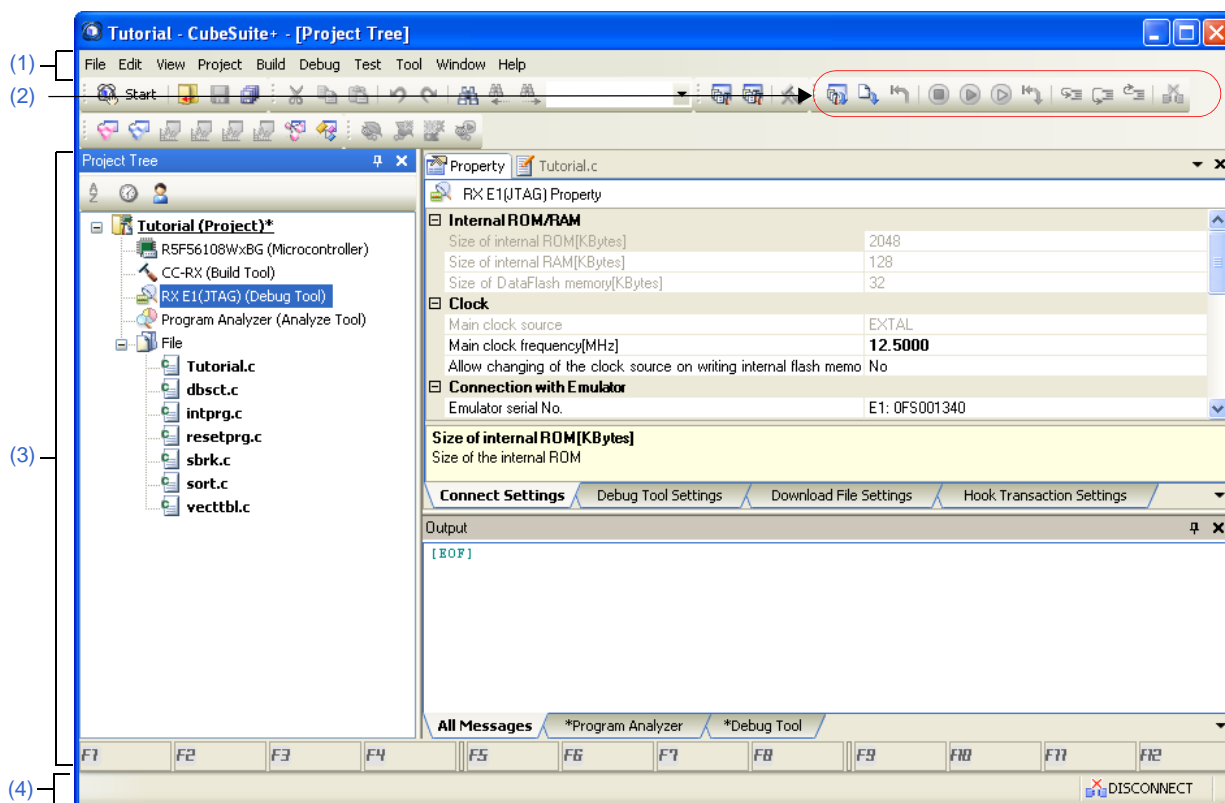
Window/Panel/Dialog Box Name	Description
Combination Condition dialog box [E1][E20]	Displays the conditions for event combination and modifies the settings.
Detailed Settings of Timer Measurement dialog box [E1] [E20]	Displays the detailed information on timer events and modifies the settings.
Detailed Settings of Execution Events dialog box	Displays the detailed information on execution type events and modifies the settings.
Detailed Settings of Access Events dialog box	Displays the detailed information on access type events and modifies the settings.
Detailed Settings of Interrupt Events dialog box [Simulator]	Displays the detailed information on interrupt events and modifies the settings.
Port Setting dialog box	Sets COM port in the Debug Console panel .
Scroll Range Settings dialog box	Sets the scroll range for the Memory panel/Disassemble panel .
Go to Line dialog box	Moves the caret to the specified line.
Go to the Location dialog box	Moves the caret to the specified position.
Data Save dialog box	Saves the settings and other data displayed in the respective windows/panels/dialogs or saves upload data.
Progress Status dialog box	Displays the progress of the processing being executed.
Option dialog box	Makes settings for various environments.
External flash memory dialog box [E1] [E20]	Selects external flash definition files (USD files).
Open Log File dialog box	Selects log files.
Select Download File dialog box	Selects files to be downloaded.
Open File dialog box	Selects files to be opened.
Print Preview window	Previews the source file before printing.
Save As dialog box	Saves files or the contents of various windows/panels/dialogs.
Select Data Save File dialog box	Selects the file to save data.

Main window

This is the first window that opens when you launch CubeSuite+.

When debugging a program, use this window to control program execution and opening of each panel.

Figure A-1. Main Window



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)

[How to open]

- Choose [All Programs] from Windows' [Start] menu and then select [Renesas Electronics CubeSuite+] and then [CubeSuite+].

[Description of each area]**(1) Menu bar**

The debug-related menu items are describe below.

Remark The items drawn from each menu can be customized using the User Settings dialog box.

(a) [View]

Each item on [View] menu and their functionality are as follows (default).

Watch	Shows the following cascaded menu to open the Watch panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Watch1	Opens the Watch panel (Watch1).
Watch2	Opens the Watch panel (Watch2).
Watch3	Opens the Watch panel (Watch3).
Watch4	Opens the Watch panel (Watch4).
Local Variable	Opens the Local Variables panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Call Stack	Opens the Call Stack panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Memory	Shows the following cascaded menu to open the Memory panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Memory1	Opens the Memory panel (Memory1).
Memory2	Opens the Memory panel (Memory2).
Memory3	Opens the Memory panel (Memory3).
Memory4	Opens the Memory panel (Memory4).
IOR	Opens the IOR panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
CPU Register	Opens the CPU Register panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Trace	Opens the Trace panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Disassemble	Shows the following cascaded menu to open the Disassemble panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Disassemble1	Opens the Disassemble panel (Disassemble1).
Disassemble2	Opens the Disassemble panel (Disassemble2).
Disassemble3	Opens the Disassemble panel (Disassemble3).
Disassemble4	Opens the Disassemble panel (Disassemble4).
Event	Opens the Events panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Debug Console	Opens the Debug Console panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.

Show Current PC Location	Displays the current PC position (PC register value) on the Editor panel . However, this item is disabled when CubeSuite+ is disconnected from the debug tool.
Back to Last Cursor Position	Moves the caret back to where it was before it jumped to a defined place (see "(4) Jump to functions " and "(4) Moving to a symbol definition part ").
Forward to Next Cursor Position	Moves the caret back to where it was before [Back to Last Cursor Position] was executed.
Tag Jump	If, on Editor panel or Output panel , there is file name or line/column information on the line at which the caret exists, this menu causes a jump to the relevant line/column in the relevant file (see "(5) Jump to a desired line (tag jump) ").

(b) [Debug]

Each item of the [Debug] menu and their functionality are as follows (default).

Download	Downloads a specified file to the debug tool currently selected in the active project. If CubeSuite+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. However, this item is disabled during program execution or when [Build & Download] is under execution.
Build & Download	Builds a project and executes a download to the debug tool currently selected in the active project after the build is complete. If CubeSuite+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. However, if a build fails, no download is executed.
Connect to Debug Tool	Connects to the debug tool currently selected in the active project. However, if already connected with the debug tool, this item is disabled.
Upload...	Opens the Data Save dialog box to save memory contents to a file. However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.
Hot Plug-in [E1(JTAG)] [E20(JTAG)]	Hot-plugs in to the debug tool currently selected in the active project in order to debug the target system currently under execution (see Section "2.4.3 Connect the debug tool to CubeSuite+ using hot plug-in [E1(JTAG)] [E20(JTAG)] "). However, if already connected with the debug tool, this item is disabled.
Disconnect from Debug Tool	Terminates communication with the currently connected debug tool. However, this item is disabled when [Build & Download] is under execution or when already disconnected from the debug tool.
Stop	Forcibly halts the currently executed program. However, this item is disabled when the program is already halted or disconnected from the debug tool.
Go	Runs the program from the current PC position and when the condition for a set break event holds true, stops the program under execution. However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.
Ignore Break and Go	Runs the program from the current PC position and continues running it ignoring the break and action events set. However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.

Step In	Executes the program step by step ^{Note} at a time, updating the content of each panel. For a function call, the program stops at the beginning of a called program. However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.
Step Over	Runs the program from the current PC position by executing one step ^{Note} at a time, updating the content of each panel. For a function call by a jump to subroutine instruction, all of the source lines or instructions in that function are executed successively in one step until a place is reached at which control returns from the function (step-executed until there is as many nesting as would be when a jump to subroutine instruction is executed). For other than a jump to subroutine instruction, the same operation as [Step In] is selected is performed. However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.
Return Out	Runs the program until control returns from the currently executed function (until control returns to the calling function) ^{Note} . However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.
CPU Reset	Resets the CPU (program is not executed). However, this item is disabled when [Build & Download] is under execution or when disconnected from the debug tool.
Restart	Resets the CPU once and then starts running the program from the reset address. However, this item is disabled when [Build & Download] is under execution or when disconnected from the debug tool.



Note Step execution can be performed either at the source level or at the instruction level.











For details, see Section "2.9.3 Execute programs in steps".

(2) Debug toolbar

The debug toolbar comprises buttons each representing a command for controlling program execution. Each button and their functionality are as follows (default).

- Remarks 1.** The buttons in each toolbar can be customized using the User Settings dialog box. Also, this same dialog box can be used to create a new toolbar.
- 2.** Right-clicking on the toolbar displays a context menu, which allows selection of a group to be displayed in or hidden from the toolbar.

	Builds a project and, after building, downloads the file to the active project's debug tool. If CubeSuite+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. However, if a build fails, no download is executed. This is the same as selecting [Build and download to the debug tool] from the [Debug] menu.
	Downloads a specified file to the active project's debug tool. If CubeSuite+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. However, this item is disabled during program execution or when [Build & Download] is under execution. This is the same as selecting [Download] from the [Debug] menu.

	<p>Resets the CPU (program is not executed).</p> <p>However, this item is disabled when [Build & Download] is under execution or when disconnected from the debug tool.</p> <p>This is the same as selecting [CPU Reset] from the [Debug] menu.</p>
	<p>Forcibly halts the currently executed program.</p> <p>However, this item is disabled when the program is already halted or disconnected from the debug tool.</p> <p>This is the same as selecting [Stop] from the [Debug] menu.</p>
	<p>Runs the program from the current PC position and when the condition for a set break event holds true, stops the program under execution.</p> <p>However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.</p> <p>This is the same as selecting [Go] from the [Debug] menu.</p>
	<p>Runs the program from the current PC position and continues running it ignoring the break and action events set.</p> <p>However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.</p> <p>This is the same as selecting [Ignore break and go] from the [Debug] menu.</p>
	<p>Resets the CPU once and then starts running the program from the reset address.</p> <p>However, this item is disabled when [Build & Download] is under execution or when disconnected from the debug tool.</p> <p>This is the same as selecting [Restart] from the [Debug] menu.</p>
	<p>Runs the program from the current PC position by executing one step^{Note} at a time, updating the content of each panel (step-in execution).</p> <p>For a function call, the program stops at the beginning of the called function.</p> <p>However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.</p> <p>This is the same as selecting [Step In] from the [Debug] menu.</p>
	<p>Runs the program from the current PC position by executing one step^{Note} at a time, updating the content of each panel (step-over execution).</p> <p>For a function call by a jump to subroutine instruction, all of the source lines or instructions in that function are executed successively in one step until a place is reached at which control returns from the function (step-executed until there is as many nesting as would be when a jump to subroutine instruction is executed).</p> <p>For other than a jump to subroutine instruction, the same operation as the  button is clicked is performed.</p> <p>However, this item is disabled during program execution, when [Build & Download] is under execution, or when disconnected from the debug tool.</p> <p>This is the same as selecting [Step Over] from the [Debug] menu.</p>
	<p>Runs the program until control returns from the currently executed function (until control returns to the calling function)^{Note} (return-out execution).</p> <p>However, this item is disabled during program execution, For other than a jump to subroutine instruction, the same operation as the when [Build & Download] is under execution, or when disconnected from the debug tool.</p> <p>This is the same as selecting [Return Out] from the [Debug] menu.</p>
	<p>Disconnects communication with the currently connected debug tool.</p> <p>However, this item is disabled when [Build & Download] is under execution or when already disconnected from the debug tool.</p> <p>This is the same as selecting [Disconnect from Debug Tool] from the [Debug] menu.</p>

Note Step execution can be performed either at the source level or at the instruction level.
For details, see Section "2.9.3 Execute programs in steps".

(3) Panel display area

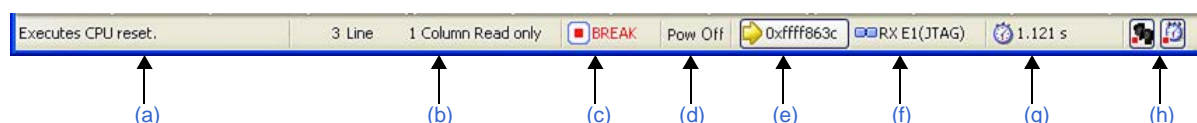
This area displays various panels available.

For details about displayed contents, see the section in which the relevant panel is described.

(4) Status bar

The status bar displays the information shown below.

Figure A-2. Statusbar



(a) Status message

Displays the following messages.

- Simple description of a selected menu item
- Message to notify that a value input in the panel/dialog is invalid
- Message to notify that the character string specified in the Find and Replace dialog box could not be found
- Message to identify the cause of program break (see Section "2.10 Stop Programs (Break)")

(b) Focus panel status information


Displays status information on the currently focused panel (e.g., information about caret position or overwrite/insertion mode).

However, this status is hidden for panels that do not have status information.

(c) Execution status

Shows the current execution status of a program using the following icons and character strings.

However, this status is hidden when CubeSuite+ is disconnected from the debug tool.

Program status	Displayed Content
Under execution	 RUN
Now halted	 BREAK
Step execution in progress	 STEP

(d) CPU status [E1(JTAG)][E20(JTAG)]

Displays the current CPU status of the debug tool.

If the CPU is in several different status at the same time, each status is shown separated by "&".

However, this status is hidden when CubeSuite+ is disconnected from the debug tool.

Debug Tool	Displayed Content	CPU Status
E1(JTAG) E20(JTAG)	Reset	Reset state
	Pow Off	Target not supplied with power
	Sleep	Now in sleep mode
	Standby	Now in standby mode

(e) Current PC position

Displays a hexadecimal value representing the current PC position.

Clicking this area moves the caret to the current PC position on the [Editor panel](#).

Placing the cursor over this area will pop up the following information.

- Current PC: 0x *current PC value (source name # number of lines*^{Note})

However, this status is hidden when CubeSuite+ is disconnected from the debug tool.



Note If this information cannot be obtained, it is substituted for by "symbol name + offset value".

Remark "In execution" message is displayed while the program is running.

When the real-time display update function is enabled, the PC position is updated and displayed at the set interval.

(f) Connection status with the debug tool

Shows the current status of connection with the debug tool using the following icons and character strings.

Connection Status	Displayed content
Currently connected	 <i>Debug tool name</i>
Currently disconnected	 Not connected

(g) Run-Break timer result

Displays the results of measurements by the Run-Break timer (see Section "2.14.2 [Measuring execution time from start to stop](#)"). The display unit is determined by the measurement results.

However, this status is hidden when CubeSuite+ is disconnected from the debug tool.






Status	Displayed content
No measurements mode	Not measured
Measurement in progress	Measurement in progress
When overflowed	OVERFLOW




(h) Debug tool status

Displays the current status of each function of the debug tool using the following icons.

When a function is not in operation, you can enable/disable it^{Note} by clicking the corresponding icon.

However, this status is hidden when CubeSuite+ is disconnected from the debug tool.

Function	In operation	Not in operation (Enable)	Disable
Trace			
Timer [E1(JTAG)] [E20(JTAG)]			-

Function	In operation	Not in operation (Enable)	Disable
Coverage [Simulator]			

Note [E20(JTAG)]

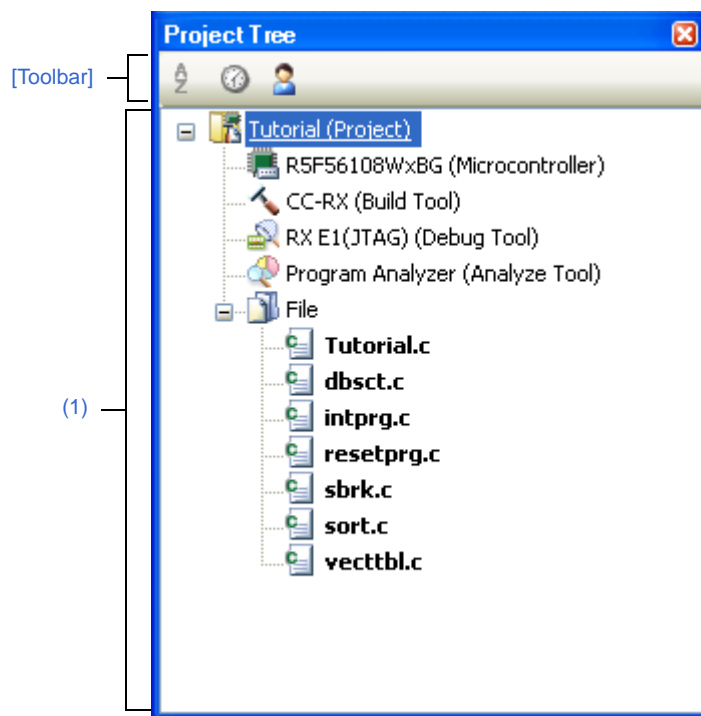
The setting of the [Usage of trace function] property in the [Trace] category on the [Property panel's \[Debug Tool Settings\] tab](#) is switched between Trace and Realtime RAM monitor.

[Simulator]

Reflected in the specification of [Use trace function] or [Use coverage function] property in the [Trace] or [Coverage] category on the [Property panel's \[Debug Tool Settings\] tab](#).

Project Tree panel

This panel shows the constituent elements of the project (e.g., microcontroller, build tool, and debug tool) in tree form. Note that the debug tools used are selected or switched from one to another on this panel.

Figure A-3. Project Tree Panel

This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[Context menu\]](#)

[How to open]

- Choose [Project Tree] from the [View] menu.

















[Description of each area]**(1) Project tree area**

This area displays the project's constituent elements in tree form at the following nodes.

Node	Description
<i>Microcontroller type</i> <i>Debug tool name</i> (debug tool)	<ul style="list-style-type: none"> - <i>Microcontroller type</i> Shows the type name of a selected microcontroller (RX). - <i>Debug tool name</i> Shows the debug tool name used in the project (E1(Serial), E1(JTAG), E20(Serial), E20(JTAG), or Simulator). When a new project is created, "Simulator" is set.

When a node is selected, its detailed information (properties) is displayed on the [Property panel](#) allowing the settings on it to be changed. (If this panel is not open yet, double-click a node to open it.)

[Toolbar]

  	Displays category nodes and file names sorted in order of name (character code). Ascending display/ descending display are switched over each time the button is clicked.	
		Indicates that displayed elements are not sorted in order of name (default).
		Indicates that elements are displayed in descending order.
		Indicates that elements are displayed in ascending order.
  	Displays category nodes and file names sorted in order of timestamp. Ascending display/ descending display are switched over each time the button is clicked.	
		Indicates that displayed elements are not sorted in order of timestamp (default).
		Indicates that elements are displayed in descending order.
		Indicates that elements are displayed in ascending order.
 	Displays category nodes and file names in the order specified by the user.	
		Indicates that elements are not displayed in customized order.
		Indicates that elements are displayed in customized order (default). Drag-and-drop a category node or file name to customize the order in which they are displayed.

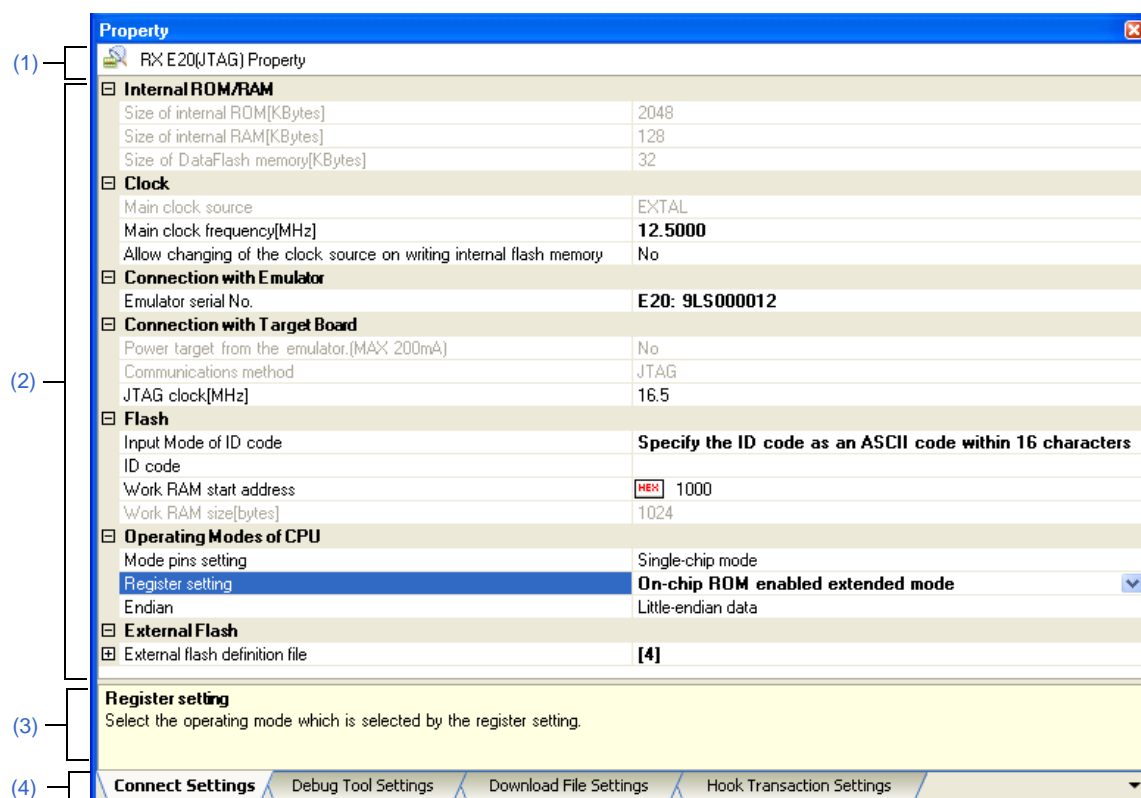
[Context menu]

Using Debug Tool	Shows a cascaded menu to select the debug tool used. Note that the debug tools displayed here differ with the microcontroller type selected for the project (see " Table 2-1. Relationship between Types of Microcontroller and Connectable Debug Tools ").
Microcontroller type E1(Serial)	Uses E1 in serial communication mode.
Microcontroller type E1(JTAG)	Uses E1 in JTAG communication mode.
Microcontroller type E20(Serial)	Uses E20 in serial communication mode.
Microcontroller type E20(JTAG)	Uses E20 in JTAG communication mode.
Microcontroller type Simulator	Uses the simulator.
Property	Displays the properties of the selected debug tool on the Property panel .

Property panel

This panel is used to display and set the debug tool operation environment that is selected in the [Project Tree panel](#).

Figure A-4. Property Panel (When E20(JTAG) Is Selected)



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[\[Edit\] menu \(Property panel-dedicated items\)\]](#)
- [\[Context menu\]](#)

[How to open]

- On the [Project Tree panel](#), select the [*Microcontroller type Debug tool name* (Debug Tool)] node to use, and then select [Property] from the [View] menu or the context menu.
- On the [Project Tree panel](#), double-click the [*Microcontroller type Debug tool name* (Debug Tool)] node to use.

Remark If this panel has been opened, the detailed information on the debug tool is displayed by selecting the [*Microcontroller type Debug tool name* (Debug Tool)] node on the [Project Tree panel](#).



[Description of each area]


(1) Selected node area

In this area, the name of the selected debug tool on the [Project Tree panel](#) is displayed.

(2) Detailed information display/change area

In this area, the detailed information on the debug tool that is selected in the [Project Tree panel](#) is displayed by category in the list. Also, you can directly change its settings.

The  mark indicates all the items in the category are expanded. The  mark indicates all the items are shrink. You can expand/shrink the items by clicking these marks or double-clicking the category name.

Note that only the hexadecimal number is allowed in the text box if the  mark is displayed in the property configuration area.

For details on the information/how to setup in the category and property items contained in it, see the section explaining the corresponding tab.

(3) Property description area

In this area, brief description of the categories and properties selected in the detailed information display/change area is displayed.

(4) Tab selection area

Categories for the display of the detailed information are changed when each tab is selected.

In this panel, following tabs are contained (see the section explaining each tab for details on the display/setting on the tab).

- [\[Connect Settings\] tab](#)
- [\[Debug Tool Settings\] tab](#)
- [\[Download File Settings\] tab](#)
- [\[Hook Transaction Settings\] tab](#)

[[Edit] menu (Property panel-dedicated items)]

Undo	Undoes the latest property value editing being done.
Cut	Deletes the selected character string(s) and copies them to the clipboard while editing the property value.
Copy	Copies the contents of the selected range to the clipboard as character string(s).
Paste	Pastes the contents of the clipboard to the property value while editing the property value.
Delete	Deletes the selected character string(s) while editing the property value.
Select All	Selects all the character strings in the selected property while editing the property value.

[Context menu]

[While not editing the property value]

Reset to Default	Restores the selected setting of the property item to default value.
Reset All to Default	Restores all the selected settings of the property items on the tab to default value.

[While editing the property value]

Undo	Undoes the latest property value editing being done.
Cut	Deletes the selected character string(s) and copies them to the clipboard while editing the property value.
Copy	Copies the contents of the selected range to the clipboard as character string(s).
Paste	Pastes the contents of the clipboard to the property value while editing the property value.
Delete	Deletes the selected character string(s) while editing the property value.
Select All	Selects all the character strings in the selected property while editing the property value.

[Connect Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.

- (1) [Internal ROM/RAM]
- (2) [Endian] [Simulator]
- (3) [Clock]
- (4) [Connection with Emulator] [E1] [E20]
- (5) [Connection with Target Board] [E1] [E20]
- (6) [Flash] [E1] [E20]
- (7) [Operating Modes of CPU] [E1] [E20]
- (8) [External Flash] [E1] [E20]
- (9) [Peripheral Function Simulation] [Simulator]

Figure A-5. Property Panel: [Connect Settings] Tab [E1(Serial)]

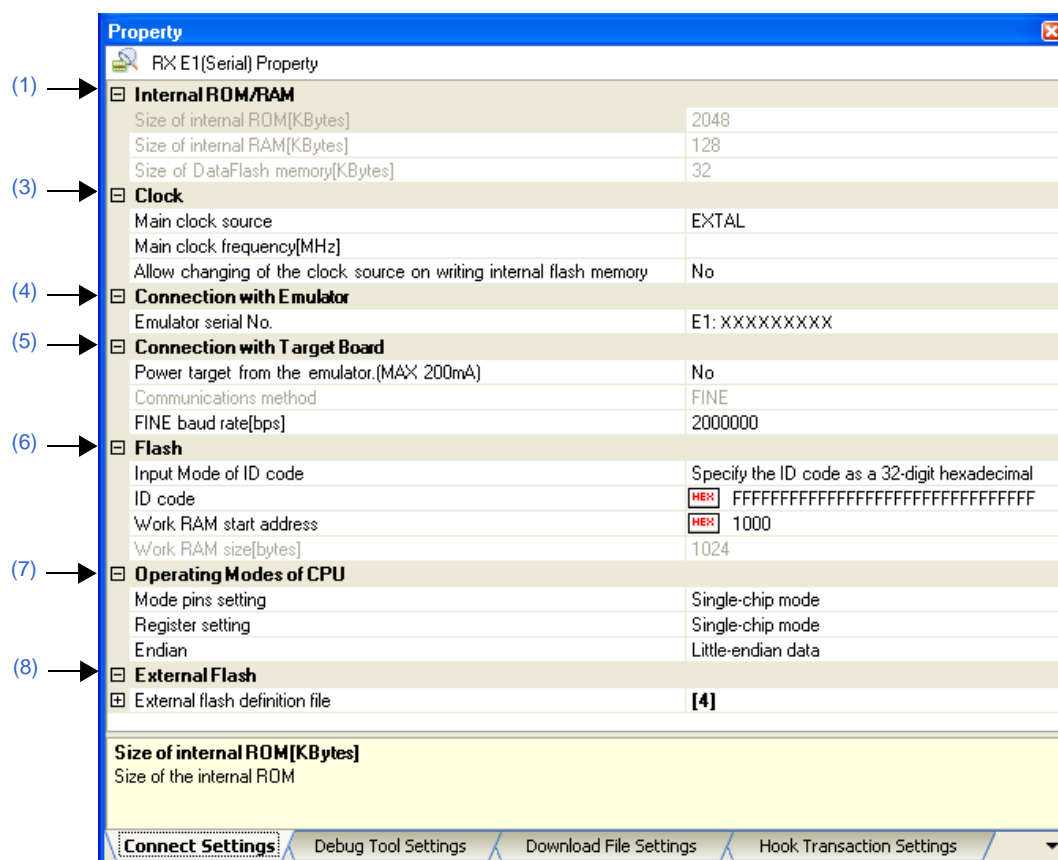


Figure A-6. Property Panel: [Connect Settings] Tab [E1(JTAG)]

Property

RX E1(JTAG) Property

(1) → **Internal ROM/RAM**

Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32

(3) → **Clock**

Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No

(4) → **Connection with Emulator**

Emulator serial No.	E1: XXXXXXXXXX
---------------------	----------------

(5) → **Connection with Target Board**

Power target from the emulator.(MAX 200mA)	No
Communications method	JTAG
JTAG clock[MHz]	16.5

(6) → **Flash**

Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024

(7) → **Operating Modes of CPU**

Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data

(8) → **External Flash**

External flash definition file	[4]
--------------------------------	-----

Internal ROM/RAM

Connect Settings | Debug Tool Settings | Download File Settings | Hook Transaction Settings

Figure A-7. Property Panel: [Connect Settings] Tab [E20(Serial)]

Property

RX E20(Serial) Property

(1) → **Internal ROM/RAM**

Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32

(3) → **Clock**

Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No

(4) → **Connection with Emulator**

Emulator serial No.	E20: XXXXXXXXXX
---------------------	-----------------

(5) → **Connection with Target Board**

Power target from the emulator.(MAX 200mA)	No
Communications method	FINE
FINE baud rate[bps]	2000000

(6) → **Flash**

Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024

(7) → **Operating Modes of CPU**

Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data

(8) → **External Flash**

External flash definition file	[4]
--------------------------------	-----

Internal ROM/RAM

Connect Settings | Debug Tool Settings | Download File Settings | Hook Transaction Settings

Figure A-8. Property Panel: [Connect Settings] Tab [E20(JTAG)]

The screenshot shows the 'Property' window for 'RX E20(JTAG) Property'. It features several expandable sections and a bottom tab bar. Numbered callouts (1-8) point to specific elements:

- (1) points to the 'Internal ROM/RAM' section.
- (3) points to the 'Clock' section.
- (4) points to the 'Connection with Emulator' section.
- (5) points to the 'Connection with Target Board' section.
- (6) points to the 'Flash' section.
- (7) points to the 'Operating Modes of CPU' section.
- (8) points to the 'External Flash' section.

The bottom tab bar includes: **Connect Settings** (selected), Debug Tool Settings, Download File Settings, and Hook Transaction Settings.

Internal ROM/RAM	
Size of internal ROM[KBytes]	2048
Size of internal RAM[KBytes]	128
Size of DataFlash memory[KBytes]	32

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No

Connection with Emulator	
Emulator serial No.	E20:XXXXXXXXXX

Connection with Target Board	
Power target from the emulator.(MAX 200mA)	No
Communications method	JTAG
JTAG clock[MHz]	16.5

Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024

Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data

External Flash	
External flash definition file	[4]

Internal ROM/RAM

Figure A-9. Property Panel: [Connect Settings] Tab [Simulator]

The screenshot shows the 'Property' window for 'RX Simulator Property'. Numbered callouts (1-9) point to specific elements:

- (1) points to the 'Internal ROM/RAM' section.
- (2) points to the 'Endian' section.
- (3) points to the 'Clock' section.
- (9) points to the 'Peripheral Function Simulation' section.

The bottom tab bar includes: **Connect Settings** (selected), Debug Tool Settings, Download File Sett..., and Hook Transaction...

Internal ROM/RAM	
Size of internal ROM[KBytes]	1032
Size of internal RAM[KBytes]	64

Endian	
Endian of CPU	Little-endian data

Clock	
System clock (ICLK) frequency[MHz]	50

Peripheral Function Simulation	
Peripheral function simulation module	[2]
Peripheral clock rate	1

Size of internal ROM[KBytes]
Displays the size of the internal ROM.

[Description of each category]**(1) [Internal ROM/RAM]**

Displays detailed information on internal ROM/RAM.

Size of internal ROM[Kbytes]	Displays the internal ROM size of a selected microcontroller.	
	Default	<ul style="list-style-type: none"> - For products with internal ROM <i>Internal ROM size of a selected microcontroller</i> - For RODROP-DOWNM-less products <i>0</i>
	How to change	Not changeable
Size of internal RAM[Kbytes]	Displays the internal RAM size of a selected microcontroller.	
	Default	<i>Internal RAM size of a selected microcontroller</i>
	How to change	Not changeable
Size of DataFlash memory[Kbytes] [E1] [E20]	Displays the size of the data flash memory area of a selected microcontroller.	
	Default	<i>Data flash memory size of a selected microcontroller</i>
	How to change	Not changeable

(2) [Endian] [Simulator]

Displays detailed information on CPU endian and modifies its settings.

Endian of CPU	Displays the microcontroller's endian. The endian information set in properties of the build tool is acquired for display here.	
	Default	<i>Little-endian data</i>
	How to change	<ul style="list-style-type: none"> - For the build & debug tool Not changeable - For the debug tool only By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	<ul style="list-style-type: none"> - One of the following as selected from the drop-down list: <i>Little-endian data or Big-endian data</i>

(3) [Clock]

The detailed information on clocks is displayed and its configuration can be changed.

Main clock source [E1] [E20]	- For products with internal HOCO Specify the main clock from the EXTAL frequency and internal HOCO.	
	- For products without internal HOCO Displays EXTAL frequency as the main clock.	
	Default	EXTAL
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
Specifiable value	EXTAL	Main clock operates as EXTAL frequency.
	HOCO	Main clock operates as internal HOCO.
Main clock frequency[MHz] [E1][E20]	Specify EXTAL frequency in MHz units. Note that this property is enabled only when [EXTAL] is specified in the [Main clock source] property.	
	Default	Blank
	How to change	By entering directly from the keyboard However, changeable only when disconnected from the debug tool
	Specifiable value	0.0001 to 99.9999 (in MHz)
Allow to change the clock source on writing internal flash memory [E1] [E20]	Specify whether or not to allow manipulation of the main clock source by debugger when internal flash memory is rewritten.	
	Default	No
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	Yes
No		Manipulation of the main clock source is not allowed.
System clock (ICLK) frequency[MHz] [Simulator]	Specify the CPU's operating clock frequency.	
	Default	- [RX600 Series] 100 - [RX200 Series] 50
	How to change	By entering directly from the keyboard However, changeable only when disconnected from the debug tool
	Specifiable value	1 to 1000 (in MHz)

(4) [Connection with Emulator] [E1] [E20]

Emulator serial No.	Select the serial No. of the emulator to be connected. <small>Note</small>	
	Default	Serial No. of the connected emulator
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	Depends on the emulator used.

Note When E1(JTAG) or E1(Serial) is selected in the debug tool used, serial Nos. of E1 emulator are enumerated. Similarly, when E20(Serial) or E20(JTAG) is selected, serial Nos. of E20 emulator are enumerated.

(5) [Connection with Target Board] [E1] [E20]

The detailed information on the connection to the target board is displayed and its configuration can be changed.

Power target from the emulator (MAX 200mA)	Specify whether to supply power to the target system from E1. The E20 does not support the power supply function. Therefore, the displayed property value is [No].	
	Default	No
	How to change	- For [E1] By selecting from the drop-down list However, changeable only when disconnected from the debug tool - For [E20] Not changeable
	Specifiable value	Yes Supplies power to the target system. No Does not supply power to the target system.
Supply voltage [E1]	Specify the power voltage supplied to the target system. This property appears only when the [Power target from the emulator (MAX 200mA)] property is set to [Yes].	
	Default	3.3V
	How to change	Select from the drop-down list. Note that changes can be made only while disconnecting from the debug tool.
	Specifiable value	- One of the following as selected from the drop-down list: Note 3.3V, 5.0V
Communication method	Displays the communication method to be connected when communication between the emulator and the CPU on target system is performed.	
	Default	- [E1(JTAG)] [E20(JTAG)] JTAG - [E1(Serial)] [E20(Serial)] FINE
	How to change	Not changeable
JTAG clock [MHz]	Specify JTAG communication speed between the emulator and the CPU on target system. Note that this property is displayed only when [JTAG] is specified in the [Communication method] property.	
	Default	16.5
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	One of the following as selected from the drop-down list: 16.5, 12.38, 6.188, 3.094, 1.547

FINE baud rate[bps]	Specify FINE communication speed between the emulator and the CPU on target system. Note that this property is displayed only when [FINE] is specified in the [Communication method] property.	
	Default	2000000
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	One of the following as selected from the drop-down list: 2000000, 750000, 500000, 250000

Note The specifiable voltage value differs with each selected microcontroller.

(6) [Flash] [E1] [E20]

The detailed information on the flash memory writing is displayed and its configuration can be changed.

Input Mode of ID code	Specify ID code input mode.		
	Default	ID code specified in 32 hexadecimal digits	
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool	
	Specifiable value	Specify the ID code as a 32-digit hexadecimal	Enter ID code in 32 hexadecimal digits.
		Specify the ID code as an ASCII code within 16 characters	Enter ID code within 16 ASCII characters.
ID code	Specify ID code that is used when code in memory is read out.		
	Default	- For [Specify the ID code as a 32-digit hexadecimal] FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF - For [Specify the ID code as an ASCII code within 16 characters] Blank	
	How to change	By entering directly from the keyboard However, changeable only when disconnected from the debug tool	
	Specifiable value	- For [Specify the ID code as a 32-digit hexadecimal] 32 hexadecimal digits - For [ID code specified within 16 characters (ASCII)] 16 ASCII code characters (if less than 16 characters, 0s are added)	
Work RAM start address	Specify the location address of the work RAM used by the debugger. Specified bytes of space from the specified work RAM location address are used by the debugger firmware. ^{Note}		
	Default	Depends on the selected microcontroller	
	How to change	By entering directly from the keyboard However, changeable only when disconnected from the debug tool	
	Specifiable value	Address value that matches the internal RAM area of the selected microcontroller	

Work RAM size[bytes]	Displays the size of the work RAM used by the debugger.	
	Default	<i>Depends on the selected microcontroller</i>
	How to change	Not changeable

Note This area can also be used by the user program since the memory contents are swapped and restored. However, you cannot designate this area as the source or destination of DMA or DTC function transfer.

(7) [Operating Modes of CPU] [E1] [E20]

Displays detailed information on the microcontroller's operation modes or changes settings made in it.

Mode pins setting	Specify the operation mode set by mode pins.	
	Default	<i>Single-chip mode</i>
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	- One of the following as selected from the drop-down list: Note 1 <i>Single-chip mode or user boot mode</i>
Register setting	Specify the operation mode set by registers.	
	Default	<i>Single-chip mode</i>
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	- One of the following as selected from the drop-down list: Note 1 <i>Single-chip mode, internal ROM-enabled extension mode or internal ROM-disabled extension mode</i>
Endian	Displays the project's endian. Note 2	
	Default	<i>Little-endian data</i>
	How to change	- For the build & debug tool Not changeable - For the debug tool only By selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	- One of the following as selected from the drop-down list: <i>Little-endian data or big-endian data</i>

- Notes 1.** The specifiable operation mode differs with each selected microcontroller.
- 2.** If a microcontroller with an MDE pin is selected, be sure that the project's endian and the microcontroller's MDE pin state match. If endians are different, the project and microcontroller cannot be connected correctly.

(8) [External Flash] [E1] [E20]

Displays detailed information on external flash or changes settings made in it.

External flash definition file	Specify an external flash definition file. The number of external flash definition files specified is displayed in properties.	
	Default	0
	How to change	<ul style="list-style-type: none"> - Main item Not changeable - Sub-items By selecting in the External flash memory dialog box [E1] [E20] The External flash memory dialog box is opened by clicking the [...] button that is displayed at the right edge in the column of this property when it is selected. (External flash definition files cannot be specified on property panel.)
	Display content	<ul style="list-style-type: none"> - Main item Number of external flash definition files - Sub-items Definition file names are displayed for each index ([1] to [4]) by clicking the "+" mark on external flash definition file.

(9) [Peripheral Function Simulation] [Simulator]

Displays detailed information on peripheral function simulation or changes settings made in it.

Peripheral function simulation module	Displays usable peripheral function simulation modules, allowing to specify whether or not to use.	
	Default	<ul style="list-style-type: none"> - Main item <Number of usable peripheral function simulation modules> - Sub-items <Peripheral function simulation module name> Not use
	How to change	<ul style="list-style-type: none"> - Main item Not changeable - Sub-items Peripheral function simulation module names not changeable Whether or not to use a peripheral function simulation module is specified by selecting from the drop-down list However, changeable only when disconnected from the debug tool
	Specifiable value	Use Peripheral function simulation module is used.
		Not Use Peripheral function simulation module is not used.
	Display content	<ul style="list-style-type: none"> - Main item Number of usable peripheral function simulation modules - Sub-items Peripheral function simulation module name Usage status of peripheral function simulation module

Peripheral clock rate	Specify the ratio of peripheral clock to internal clock (how many internal clocks one peripheral clock is equivalent).	
	Default	1
	How to change	By selecting from the drop-down list
	Specifiable value	- One of the following as selected from the drop-down list: 1, 2, 3, 4, 6, 8, 12, 16, 24, 32, 64

[Debug Tool Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.

- (1) [Memory]
- (2) [Access Memory While Running]
- (3) [Break] [E1] [E20]
- (4) [System] [E1] [E20]
- (5) [Trace]
- (6) [Timer] [E1][E20]
- (7) [Coverage] [Simulator]
- (8) [Stream I/O] [Simulator]
- (9) [Execution Mode] [Simulator]
- (10) [Instruction Decode Cache] [Simulator]

Figure A-10. Property Panel: [Debug Tool Settings] Tab [E1(Serial) [RX600 Series]]

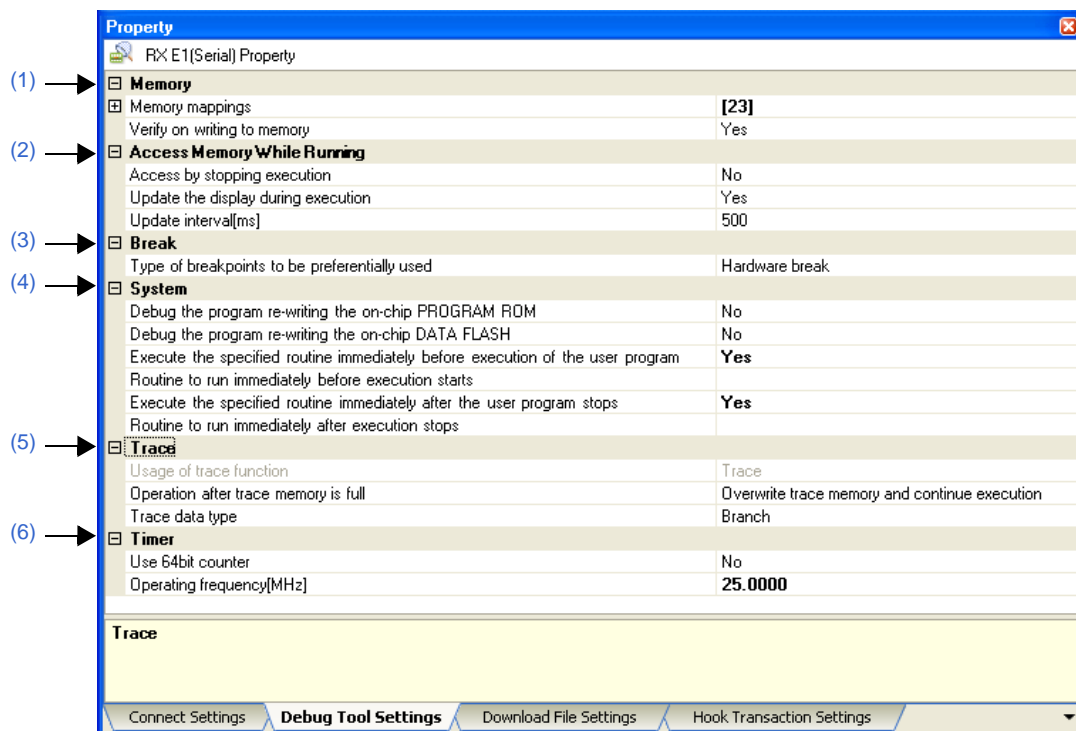


Figure A-11. Property Panel: [Debug Tool Settings] Tab [E1(Serial) [RX200 Series]]

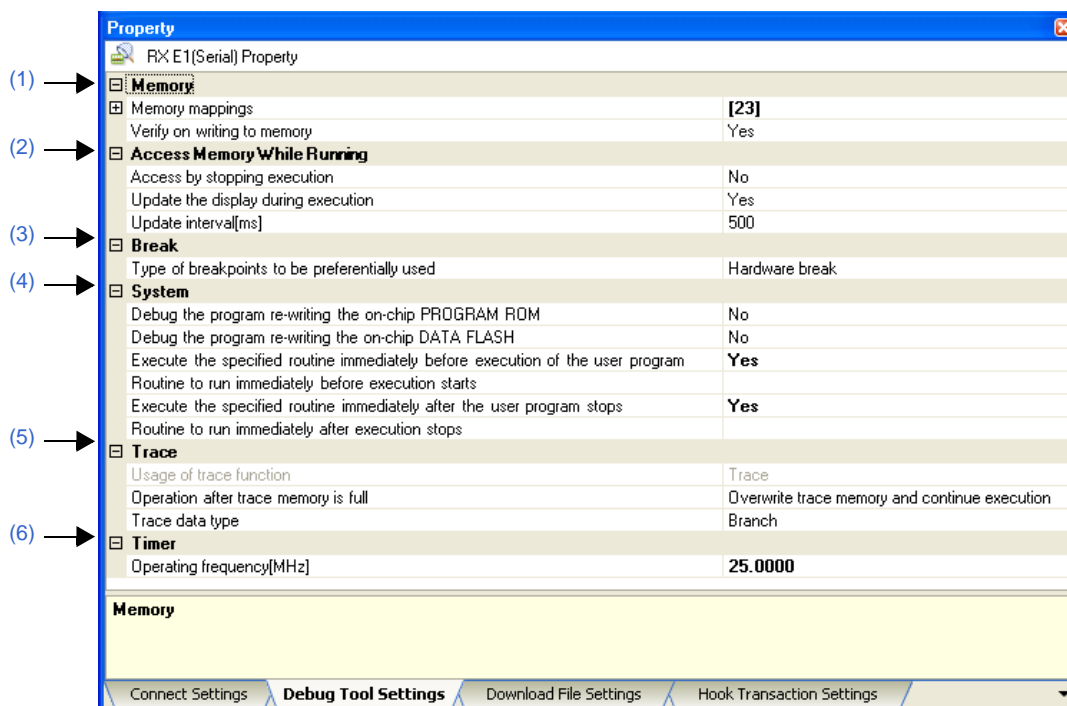


Figure A-12. Property Panel: [Debug Tool Settings] Tab [E1(JTAG) [RX600 Series]]

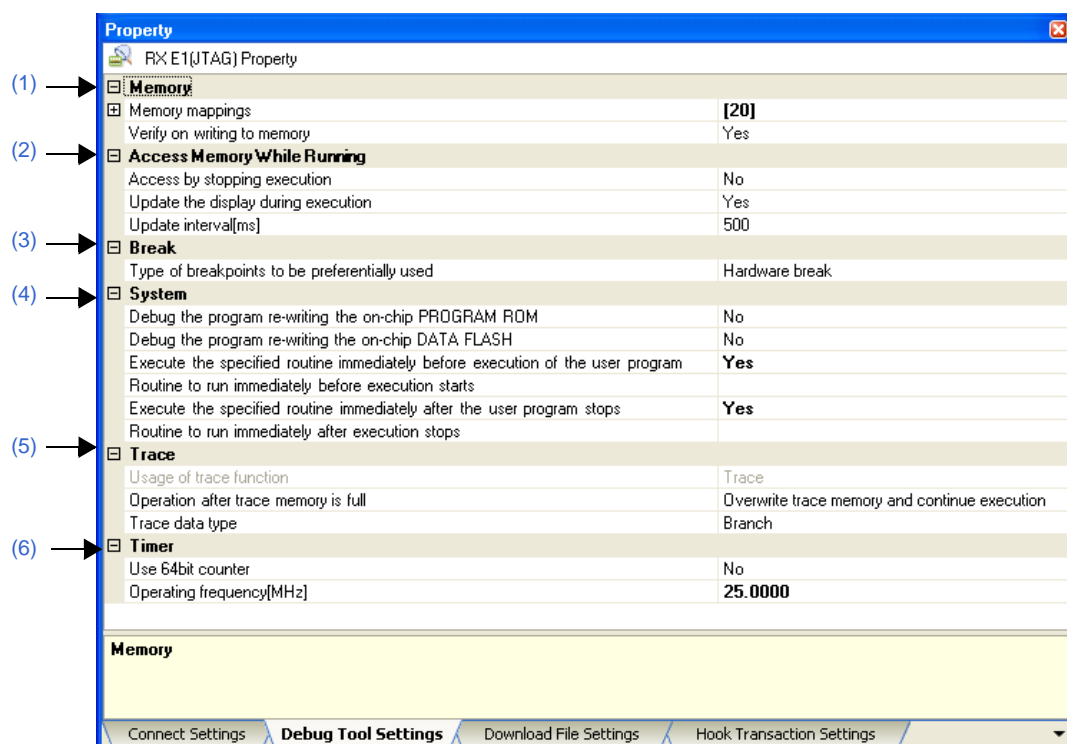


Figure A-13. Property Panel: [Debug Tool Settings] Tab [E20(Serial) [RX600 Series]]

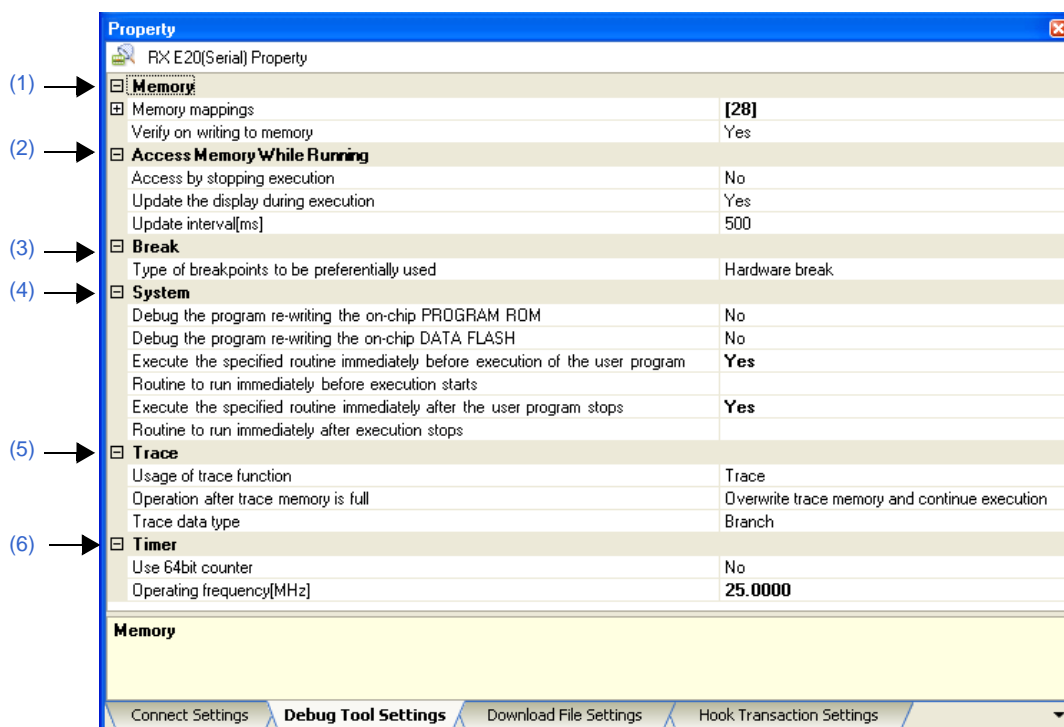


Figure A-14. Property Panel: [Debug Tool Settings] Tab [E20(Serial) [RX200 Series]]

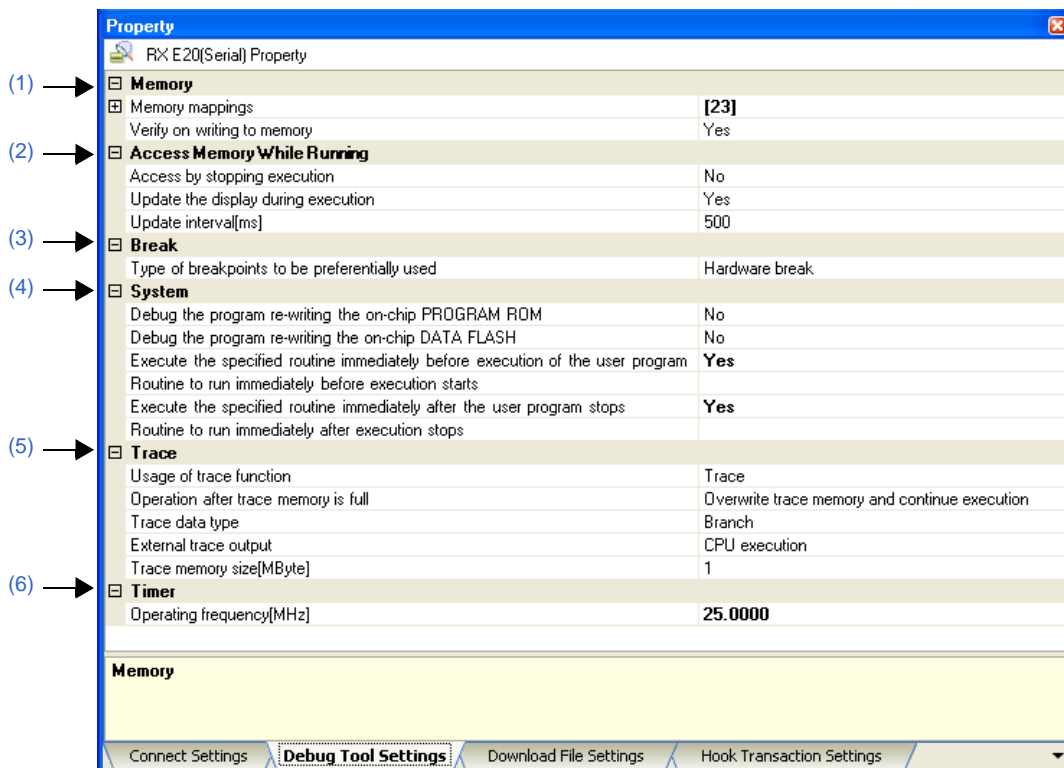


Figure A-15. Property Panel: [Debug Tool Settings] Tab [E20(JTAG) [RX600 Series]]

Property ✕

RX E20(JTAG) Property

(1) → **Memory**

Memory mappings	[28]
Verify on writing to memory	Yes

(2) → **Access Memory While Running**

Access by stopping execution	No
Update the display during execution	Yes
Update interval[ms]	500
Enable the automatic update of realtime display	Yes

(3) → **Break**

Type of breakpoints to be preferentially used	Hardware break
---	----------------

(4) → **System**

Debug the program re-writing the on-chip PROGRAM ROM	No
Debug the program re-writing the on-chip DATA FLASH	No
Execute the specified routine immediately before execution of the user program	Yes
Routine to run immediately before execution starts	
Execute the specified routine immediately after the user program stops	Yes
Routine to run immediately after execution stops	

(5) → **Trace**

Usage of trace function	Real-time RAM monitor
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Data access
External trace output	CPU execution
Trace memory size[MByte]	1

(6) → **Timer**

Use 64bit counter	No
Operating frequency[MHz]	25.0000

Memory

Connect Settings | **Debug Tool Settings** | Download File Settings | Hook Transaction Settings

Figure A-16. Property Panel: [Debug Tool Settings] Tab [Simulator [RX600 Series]]

Property ✕

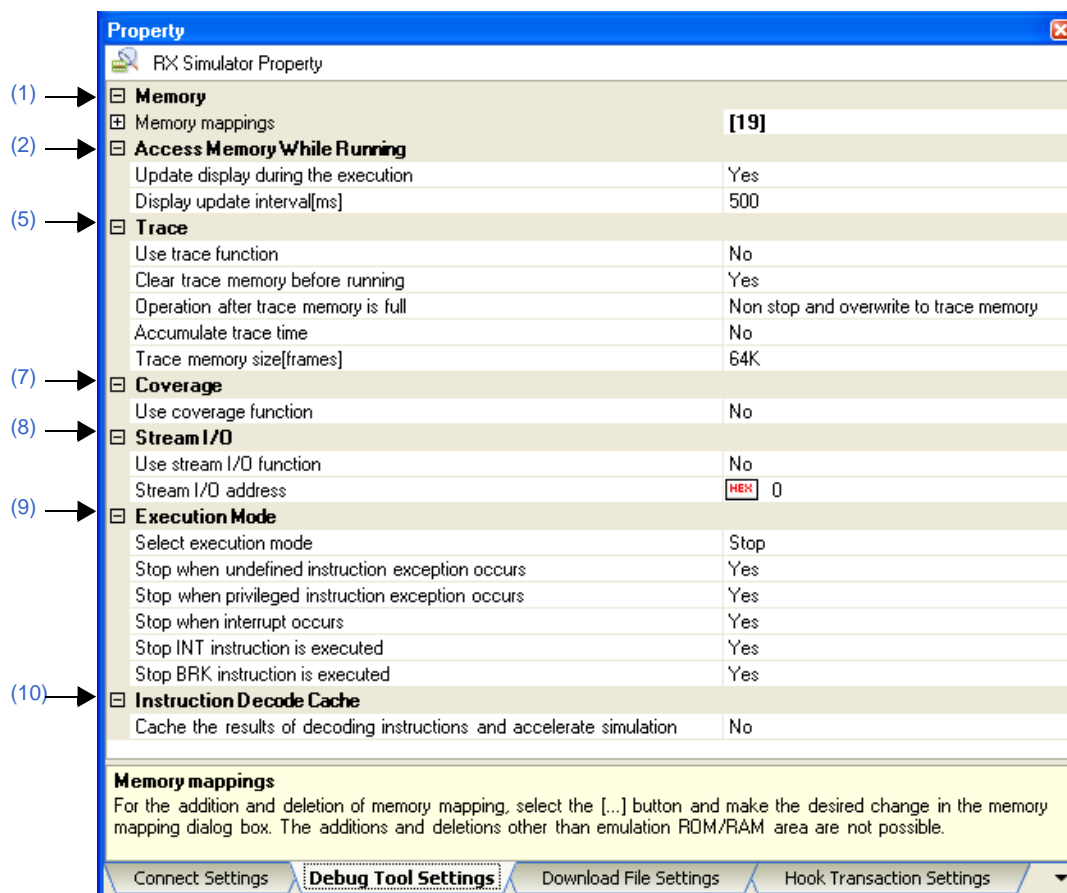
RX Simulator Property

(1) →	Memory	
	Memory mappings	[19]
(2) →	Access Memory While Running	
	Update display during the execution	Yes
	Display update interval[ms]	500
(5) →	Trace	
	Use trace function	No
	Clear trace memory before running	Yes
	Operation after trace memory is full	Non stop and overwrite to trace memory
	Accumulate trace time	No
	Trace memory size[frames]	64K
(7) →	Coverage	
	Use coverage function	No
(8) →	Stream I/O	
	Use stream I/O function	No
	Stream I/O address	HEX 0
(9) →	Execution Mode	
	Select execution mode	Stop
	Stop when undefined instruction exception occurs	Yes
	Stop when privileged instruction exception occurs	Yes
	Stop when access exception occurs	Yes
	Stop when floating-point exception occurs	Yes
	Stop when interrupt occurs	Yes
	Stop INT instruction is executed	Yes
	Stop BRK instruction is executed	Yes
(10) →	Instruction Decode Cache	
	Cache the results of decoding instructions and accelerate simulation	No

Memory

Connect Settings | **Debug Tool Settings** | Download File Settings | Hook Transaction Settings | ▼

Figure A-17. Property Panel: [Debug Tool Settings] Tab [Simulator [RX200 Series]]

**[Description of each category]****(1) [Memory]**

The detailed information on memories is displayed and its configuration can be changed.

For details on memory types that are displayed, see the "[Memory Mapping dialog box \[Simulator\]](#)".

Memory mappings	Current memory mapping status is displayed by the types of memory area Note 1 .	
	Default	[Total number of the memory mapping types]
	How to change	<p>- [Simulator]</p> <p>Specify with the Memory Mapping dialog box [Simulator].</p> <p>The Memory Mapping dialog box is opened when clicking the [...] button appears at right by selecting the mapping value (you cannot change the mapping value on this panel).</p>
	Display Contents	<p>Displays the memory mapping status by the types of memory area.</p> <p>The following detailed information is displayed by clicking the "+" mark of each memory type.</p> <ul style="list-style-type: none"> - Memory type - Start address - End address - Access width[bits] [E1] [E20] Note 2 - Endian Note 2 Note 3

Verify on writing to memory [E1] [E20]	Specify whether to perform a verify check when the memory value is initialized.		
	Default	Yes	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Executes the verify check.
		No	Does not execute the verify check.

Notes 1. This refers to the type of memory-mapped areas registered in a device file.

2. The access width and the endian can only be changed when [Memory type] is an external area and the debug tool is disconnected. Also, the specifiable value differs with each selected microcontroller.

3. The endian information differs in displayed contents and specifiable values depending on [Memory Type].

- External area [E1] [E20]

Specify one of the following by selecting from the drop-down list:

Same as MCU endian or different from MCU endian

- I/O register area

One of the following is displayed:

Little-endian data or big-endian data

- Emulation ROM area and emulation RAM area [Simulator]

Specify one of the following:

Little-endian data or big-endian data

- Areas other than the above

No endian information is displayed.

(2) [Access Memory While Running]

The detailed information on memory accesses while executing a program (real-time display update function: see "(4) [Displaying and changing memory contents during program execution](#)") is displayed and its configuration can be changed.

Access by stopping execution [E1] [E20]	For a memory area not accessible during program execution, specify whether access to the area is permitted.		
	Default	No	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Temporary stops execution and read/write.
		No	Does not access to the memory during program execution.
Update display during the execution	Specify whether to update the display in the Watch panel/Memory panel during a program execution.		
	Default	Yes	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Updates the display during program execution.
		No	Does not update the display during program execution.

Display update interval[ms]	Specify the interval in 100ms unit to update the contents in the Watch panel/Memory panel display while executing a program. This property appears only when the [Update display during the execution] property is set to [Yes].			
	Default	500		
	How to change	By entering directly from the keyboard		
	Specifiable value	Integer number between 100 and 65500 (rounding up the fractions less than 100 ms).		
Enable the automatic update of realtime display [E20(JTAG) [RX600 Series]]	Specify whether RRM area is automatically set. Note that this property is displayed only when [Realtime RAM Monitor] is specified in the [Usage of trace function] property and also [Yes] is specified in the [Update display during the execution] property.			
	Default	Yes		
	How to change	By selecting from the drop-down list		
	Specifiable value	Yes	Automatically sets the real-time display update function.	
		No	Does not set the real-time display update function.	

(3) [Break] [E1] [E20]

The detailed information on break functions is displayed and its configuration can be changed.

First using type of breakpoint [E1] [E20]	Specify the type of the breakpoint to use with priority when setting it at the source line or the execution address with a one click operation of the mouse in the Editor panel/Editor panel .		
	Default	Hardware break	
	How to change	By selecting from the drop-down list	
	Specifiable value	Software break	Sets software breakpoint with priority.
		Hardware break	Sets hardware breakpoint with priority.

(4) [System] [E1] [E20]

Displays detailed information on emulation system or changes settings made in it.

Debugging the program re-writing the on-chip PROGRAM ROM	Specify whether or not to debug a program that involves rewriting the internal program ROM (e.g., a program making use of ROM P/E mode).		
	Default	No	
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool	
	Specifiable value	Yes	A program that involves rewriting the internal program ROM is debugged.
		No	A program that involves rewriting the internal program ROM is not debugged.

Debugging the program re-writing the on-chip DATA FLASH	Specify whether or not to debug a program that involves rewriting the internal data flash (e.g., a program making use of data flash P/E mode).			
	Default	No		
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool		
	Specifiable value	Yes	A program that involves rewriting the internal data flash is debugged.	
		No	A program that involves rewriting the internal data flash is not debugged.	
Execute the specified routine immediately before execution of the user program	Specify whether or not to run a specified routine immediately before program execution.			
	Default	No		
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted		
	Specifiable value	Yes	A specified routine is run immediately before program execution.	
		No	A specified routine is not run immediately before program execution.	
The routine to run immediately before starting execution	Specify the address to be executed immediately before program execution. This property is displayed only when [Yes] is specified in the [Execute the specified routine immediately before execution of the user program] property.			
	Default	Blank		
	How to change	By entering directly from the keyboard However, changeable only when program execution is halted		
	Specifiable value	Depends on selected microcontroller		
Execute the specified routine immediately after halting of the user program	Specify whether or not to run a specified routine immediately after the program breaks.			
	Default	No		
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted		
	Specifiable value	Yes	A specified routine is run immediately program breaks.	
		No	A specified routine is not run immediately after program breaks.	
The routine to run immediately after halting execution	Specify the address to be executed immediately after the program breaks. This property is displayed only when [Yes] is specified in the [Execute the specified routine immediately after halting of the user program] property.			
	Default	Blank		
	How to change	By entering directly from the keyboard However, changeable only when program execution is halted		
	Specifiable value	Depends on selected microcontroller		

(5) [Trace]

The detailed information on trace functions is displayed and its configuration can be changed.

Caution [E20(JTAG) [RX600 Series]]

The trace function and the realtime RAM monitor function (RRM function) in part are usable exclusively of each other.

Usage of trace function [E1] [E20]	Specify whether the trace function is used as realtime RAM monitor function (RRM function).			
	Default	Trace		
	How to change	- [E1(Serial)] [E1(JTAG)] [E20(Serial)] Not changeable - [E20(JTAG)] By selecting from the drop-down list However, changeable only when program execution is halted		
	Specifiable value	Trace	Trace function is used preferentially.	
Realtime RAM monitor		Realtime RAM monitor function (RRM function) is used preferentially.		
Use trace function [Simulator]	Specify whether to use the trace function Note 1 .			
	Default	No		
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted		
	Specifiable value	Yes	Uses trace functions.	
No		Does not use trace functions.		
Clear trace memory before running [Simulator]	Specify trace acquisition mode.			
	Default	Yes		
	How to change	By selecting from the drop-down list		
	Specifiable value	Yes	Clears the trace memory.	
No		Does not clear the trace memory.		
Operation after trace memory is full	Specify the operation after the trace memory is full with the collected trace data.			
	Default	Non stop and overwrite to trace memory		
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted		
	Specifiable value	Non stop and overwrite to trace memory	Continues overwriting trace data even after trace memory is used up.	
		Stop trace [E1] [E20]	Stops overwriting trace data when trace memory is used up.	
Stop		Stops running the program and overwriting trace data when trace memory is used up.		
Trace data type [E1] [E20]	Specify the type of data to be acquired from trace.			
	Default	Branch		
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted		
	Specifiable value	One of the following as selected from the drop-down list: - RX600 Series Branch, branch + data access, or data access - RX200 Series Branch or data access		

External trace output [E20(JTAG) [RX600 Series]]	Specify the method for trace output to external device.		
	Default	CPU execution	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	CPU execution	CPU execution has priority over trace output.
		Trace output	Trace output has priority over CPU execution.
	Do not output	Trace information is not output.	
Trace memory size[MByte] [E20(JTAG) [RX600 Series]]	Specify the size of trace memory.		
	Default	1	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	One of the following as selected from the drop-down list: 1, 2, 4, 8, 16, 32	
Accumulate trace time [Simulator]	Specify whether to display the accumulated tracing time in the Trace panel .		
	Default	No	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Displays the accumulated tracing time.
		No	Displays the trace time with differential value.
Trace memory size[frames] [Simulator]	Specify the size of memory that holds trace data by a number of trace frames Note 2 .		
	Default	64K	
	How to change	By selecting from the drop-down list	
	Specifiable value	One of the following as selected from the drop-down list: 64K, 128K, 256K, 512K, 1M	

Notes 1. This property is automatically set to [Yes] when selecting [Start Tracing]/[Stop Tracing] from the context menu in the [Editor panel/Disassemble panel](#).

2. The trace frame is a unit for the trace data. Each fetch/write/read uses one trace frame.

(6) [Timer] [E1][E20]

The detailed information on timer functions is displayed and its configuration can be changed.

Use 64bit counter [E20(JTAG) [RX600 Series]]	Specify whether or not to use a 64-bit measurement counter. However, if [Yes] is specified, measurement is performed in only one section.		
	Default	No	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	One 64-bit measurement counter is used.
		No	Two 32-bit measurement counters are used.

Operating frequency[MHz]	Specify the operating clock frequency that is referenced when count values are converted into time.	
	Default	<i>Blank</i>
	How to change	By selecting from the drop-down list
	Specifiable value	0.0001 to 999.999 (in MHz)

(7) [Coverage] [Simulator]

The detailed information on coverage functions is displayed and its configuration can be changed.

Use coverage function	Specify whether to use the coverage function.		
	Default	<i>No</i>	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Use coverage functions
		No	Does not use coverage functions
Reuse coverage result	Specify whether to load/save the coverage measurement result when connecting to or disconnecting from the debug tool. This property appears only when the [Use coverage function] property is set to [Yes].		
	Default	<i>No</i>	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Loads/saves the coverage measurement result.
		No	Does not load/save the coverage measurement result.

(8) [Stream I/O] [Simulator]

Use stream I/O function	Specify whether or not to use the stream input/output function.		
	Default	<i>No</i>	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stream input/output function is used.
		No	Stream input/output function is not used.
Stream I/O address	Specify the start position of a stream input/output that is used to perform standard input/output or file input/output from the user program.		
	Default	<i>0</i>	
	How to change	By entering directly from the keyboard However, changeable only when program execution is halted	
	Specifiable value	Hexadecimal number in the range 0x0 to 0xFFFFFFFF	

(9) [Execution Mode] [Simulator]

Select execution mode	Specify whether or not to stop execution of the user program when a simulation error or an exception occurs.		
	Default	Stop	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Stop	Stops simulation.
		Continue	Continues simulation.
Stop when undefined instruction exception occurs	Specify whether or not to stop execution of the user program when an undefined instruction exception occurs. This property is displayed only when [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.
Stop when privileged instruction exception occurs	Specify whether or not to stop execution of the user program when a privileged instruction exception occurs. This property is displayed only when [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.
Stop when access exception occurs	Specify whether or not to stop execution of the user program when an access exception occurs. This property is displayed only when there is MPU module in the [Peripheral function simulation module] property and also [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.
Stop when floating-point exception occurs [RX600 Series]	Specify whether or not to stop execution of the user program when a floating-point exception occurs. This property is displayed only when [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.

Stop when interrupt occurs	Specify whether or not to stop execution of the user program when an interrupt occurs. This property is displayed only when [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.
Stop INT instruction is executed	Specify whether or not to stop execution of the user program when an NT instruction is executed. This property is displayed only when [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.
Stop BRK instruction is executed	Specify whether or not to stop execution of the user program when a BRK instruction is executed. This property is displayed only when [Stop] is specified in the [Select execution mode] property.		
	Default	Yes	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stops simulation.
		No	Does not stop simulation.

(10)[Instruction Decode Cache] [Simulator]

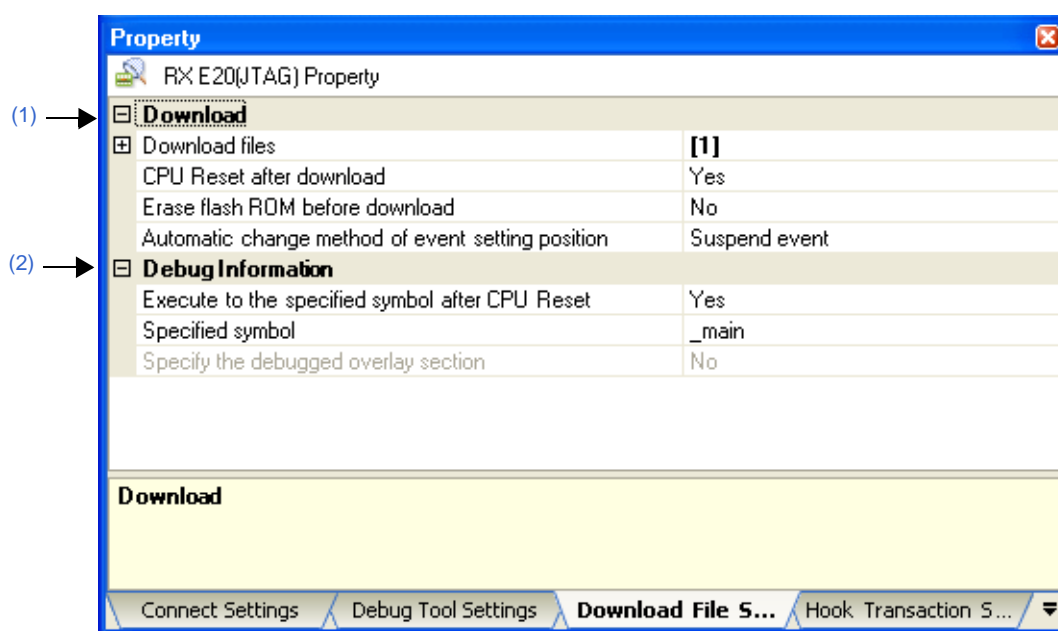
Cache the results of decoding instructions and accelerate simulation	Specify whether or not to use the instruction decode cache function.		
	Default	No	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Uses the decode cache function.
		No	Does not use the decode cache function.

[Download File Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.
For details on the download function, see "2.5 Download and Upload".

- (1) [Download]
(2) [Debug Information]

Figure A-18. Property Panel: [Download File Settings] Tab

**[Description of each category]****(1) [Download]**

The detailed information on download is displayed and its configuration can be changed.

Download files	Specify the file to download Note 1 . The names of files to be downloaded and the download conditions are listed in the lower area.			
	Default	[Number of files to download]		
	How to change	Specify with the Download Files dialog box . The Download Files dialog box is opened when clicking the [...] button appears at right by selecting this property (you cannot specify the file to download on this panel).		
CPU Reset after download	Specify whether to reset the CPU after downloading.			
	Default	Yes		
	How to change	By selecting from the drop-down list		
	Specifiable value	Yes	Resets the CPU after downloading.	
		No	Does not reset the CPU after downloading.	

Erase flash ROM before download [E1] [E20]	Specify whether to erase the flash ROM before downloading.		
	Default	No	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Erases the flash ROM before downloading.
		No	Does not erase the flash ROM before downloading.
Automatic change method of event setting position	Specify how to perform the setting again if the file is downloaded again, and the location (address) set for the currently set event changes to midway in the instruction Note 2 .		
	Default	Suspend event	
	How to change	By selecting from the drop-down list	
	Specifiable value	Move to the head of instruction	Sets the event to the top address of the instruction.
		Suspend event	Disables the event (suspended state).

Notes 1. Files specified as build targets in a main project or sub-project cannot be deleted from the target files to download (These files are automatically registered as download files by default).

See "[Table 2-2. Downloadable File Formats](#)" for downloadable file format.

- 2.** This property setting works only for the location setting of events without the debug information. The location setting of events with the debug information is always moved to the beginning of the source text line.

(2) [Debug Information]

The detailed information on debugging is displayed and its configuration can be changed.

Execute to the specified symbol after CPU Reset	Specify whether to execute the program to the specified symbol position after CPU reset.		
	Default	Yes	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Executes the program to the specified symbol position after CPU reset.
		No	Does not execute the program after CPU reset.
Specified symbol	Specify the position at which the program is stop after CPU reset. This property appears only when the [Execute to the specified symbol after CPU Reset] property is set to [Yes].		
	Default	_main	
	How to change	By entering directly from the keyboard	
	Specifiable value	Address expression from 0 to the "end address of the address space".	

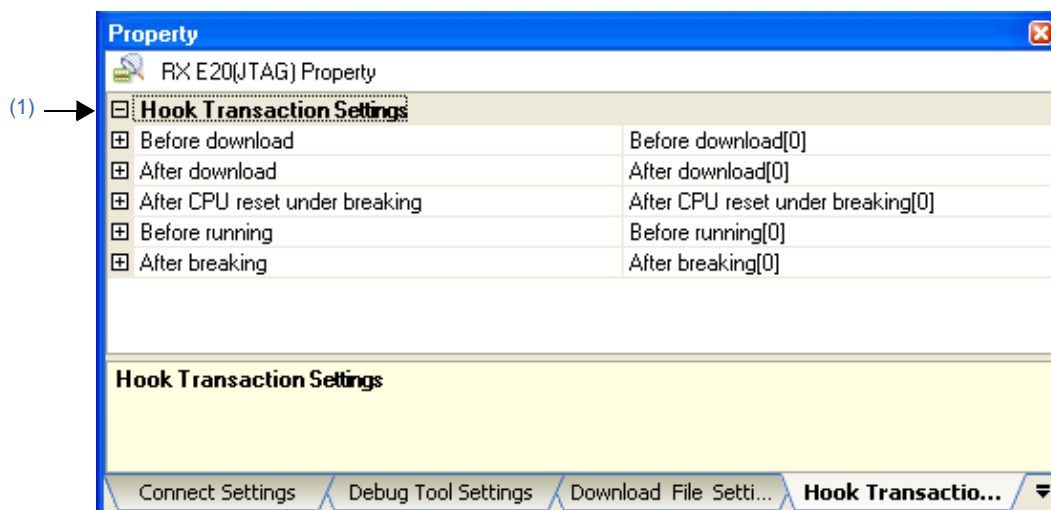
Specify the debugged overlay section	If the downloaded load module has overlay sections in it, select the section to debug.	
	Default	<ul style="list-style-type: none"> - When the load module has no overlay sections No - When the load module has overlay sections Yes
	How to change	Not changeable
Overlay sections	Displays address groups in which overlay sections are present. From the overlay sections defined in each address group, select the section to debug (priority section). Note that this property is displayed only when [Yes] is specified in the [Specify the debugged overlay section] property item.	
	Default	None
	How to change	<ul style="list-style-type: none"> - [File], [Start address], [End address] items Not changeable - [Priority section] item By selecting from the drop-down list
	Display content	Clicking the "+" mark on any address group number displays the following detailed information. The section name selected in the [Priority Section] item is the section to be debugged. <ul style="list-style-type: none"> - File - Start address - End address - Priority section

- Cautions 1.** The contents of overlay section-related settings are not saved in a project file. After downloading the load module, reset the section to be debugged.
- 2.** The information changed by an alteration of the [Priority Section] item is only debug information. The debugger does not copy data of the subject section.

[Hook Transaction Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed. For details on the hook transaction, see "2.18 Setting Up the Hook Process".

(1) [Hook Transaction Settings]

Figure A-19. Property Panel: [Hook Transaction Settings] Tab**[Description of each category]****(1) [Hook Transaction Settings]**

The detailed information on the hook transaction is displayed and its configuration can be changed.

Note that the properties on this tab can be specified via the [Text Edit dialog box](#), which is opened by clicking the [...] button that appears at right by selecting each property (you cannot specify the process directly on this panel).

Caution Up to 64 characters for one process, and up to 128 processes for each property can be set (one line in the [Text](#) area in the [Text Edit dialog box](#) is equivalent to one processing).

Before download	Specify the process to proceed right before downloading.	
	Default	<i>Before download[0]</i> ("[]" is the current number of specified processes.)
	How to change	Specify with the Text Edit dialog box .
	Format	Either one of the following - I/O register name + space + Value [Process] Automatically overwrites the value of I/O register with Value. - CPU register name + space + Value [Process] Automatically overwrites the value of CPU register with Value. - source Python script path [Process] Automatically executes a script file specified with Python script path.

After download	Specify the process to proceed right after downloading.	
	Default	<i>After download[0]</i> ("[]" is the current number of specified processes.)
	How to change	Specify with the Text Edit dialog box .
	Format	Either one of the following - I/O register name + space + Value [Process] Automatically overwrites the value of I/O register with Value. - CPU register name + space + Value [Process] Automatically overwrites the value of CPU register with Value. - source Python script path [Process] Automatically executes a script file specified with Python script path.
After CPU reset under breaking	Specify the process to proceed right after CPU reset during break.	
	Default	<i>After CPU reset under breaking[0]</i> ("[]" is the current number of specified processes.)
	How to change	Specify with the Text Edit dialog box .
	Format	Either one of the following - I/O register name + space + Value [Process] Automatically overwrites the value of I/O register with Value. - CPU register name + space + Value [Process] Automatically overwrites the value of CPU register with Value. - source Python script path [Process] Automatically executes a script file specified with Python script path.
Before running	Specify the process to proceed right before execution.	
	Default	<i>Before running[0]</i> ("[]" is the current number of specified processes.)
	How to change	Specify with the Text Edit dialog box .
	Format	Either one of the following - I/O register name + space + Value [Process] Automatically overwrites the value of I/O register with Value. - CPU register name + space + Value [Process] Automatically overwrites the value of CPU register with Value. - source Python script path [Process] Automatically executes a script file specified with Python script path.
After breaking	Specify the process to proceed right after the break.	
	Default	<i>After breaking[0]</i> ("[]" is the current number of specified processes.)
	How to change	Specify with the Text Edit dialog box .
	Format	Either one of the following - I/O register name + space + Value [Process] Automatically overwrites the value of I/O register with Value. - CPU register name + space + Value [Process] Automatically overwrites the value of CPU register with Value. - source Python script path [Process] Automatically executes a script file specified with Python script path.

Editor panel

This panel is used to display and edit text files and source files.

Furthermore, the source level debugging (see "2.9.3 Execute programs in steps") and the code coverage measurement result display **[Simulator]** (see "2.15 Measure Coverage [Simulator]") can be performed when connected to the debug tool and the downloaded source file is opened in this panel.

When opened the file encoding (Shift_JIS/EUC-JP/UTF-8) and newline code is automatically detected and retained when it is saved. You can open a file with a specific encoding selected in the [Encoding dialog box](#). If the encoding and newline code is specified in the [Save Settings dialog box](#) then the file is saved with those settings.

This panel can be opened multiple times (max. 100 panels).

- Remarks 1.** When a project is closed, all of the Editor panels displaying a file being registered in the project are closed.
- 2.** When a file is excluded from a project, the Editor panel displaying the file is closed.
- 3.** A message is shown when the downloaded load module file is older than the source file to be opened. This is due to the debug information not matching the source code being viewed.

Figure A-20. Editor Panel

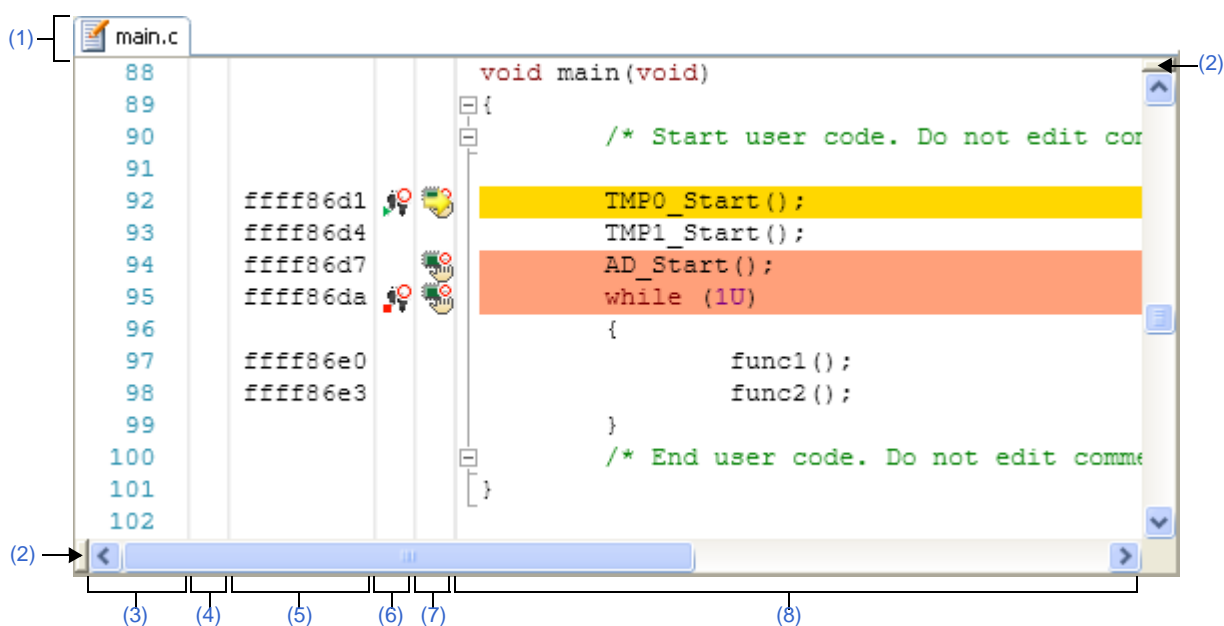
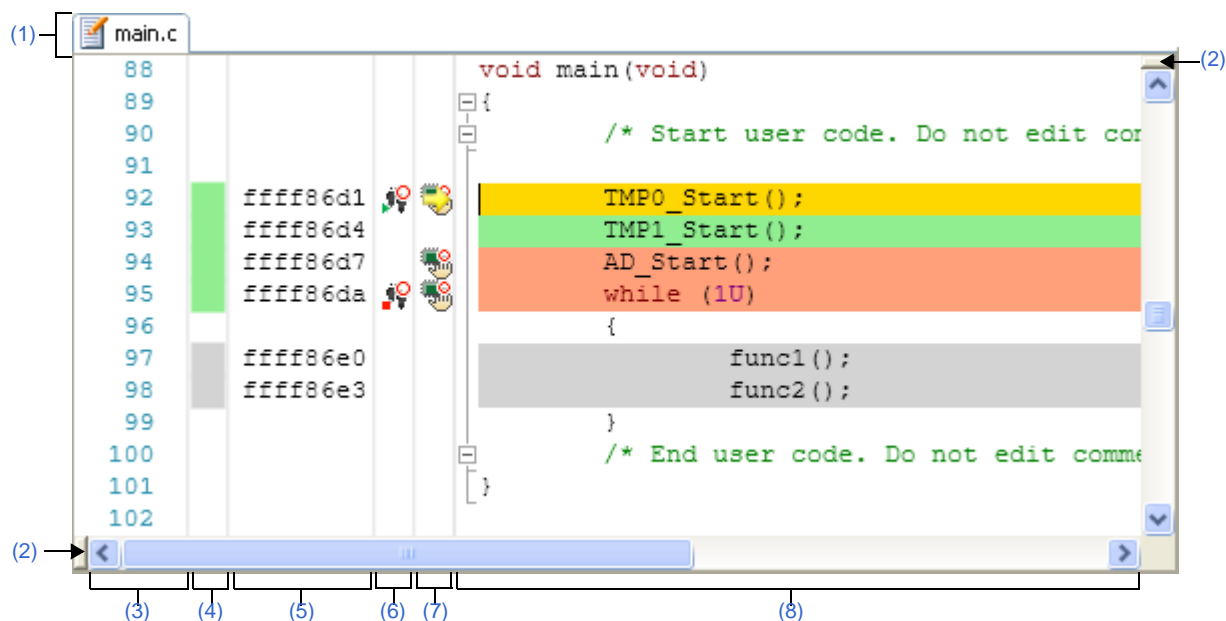


Figure A-21. Editor Panel (When Code Coverage Measurement Result Is Displayed)



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[\[File\] menu \(Editor panel-dedicated items\)\]](#)
- [\[\[Edit\] menu \(Editor panel-dedicated items\)\]](#)
- [\[\[Window\] menu \(Editor panel-dedicated items\)\]](#)
- [\[Context menu\]](#)

[How to open]

- Automatically opens after downloading the load module file with debug information.
- On the [Project Tree panel](#), double-click the file.
- On the [Project Tree panel](#), select a source file, and then select [Open] from the context menu.
- On the [Project Tree panel](#), select [Add] >> [Add New File...] from the context menu, and then create a text file or source file.
- On the [Disassemble panel](#), [Call Stack panel](#), [Trace panel](#), or [Events panel](#), select [Jump to Source] from the context menu.
- Automatically opens if there is a source text line correspond to the current PC value when the current PC value is forcibly changed or the program stops executing.

[Description of each area]**(1) Titlebar**

The name of the opened text file or source file is displayed.

Marks displayed at the end of the file name indicate the following:

Mark	Description
*	The text file has been modified since being opened.
!	Update time and date of the source file opened are later than the one of the downloaded load module file in the case the text file, which is contained in the load module file downloaded in the connected debug tool as debug information, is opened.
(Read only)	The opened text file is read only.

When you right-click in this area, the context menu is displayed. The following menu items are exclusive for the Editor panel (other items are common to all the panels).

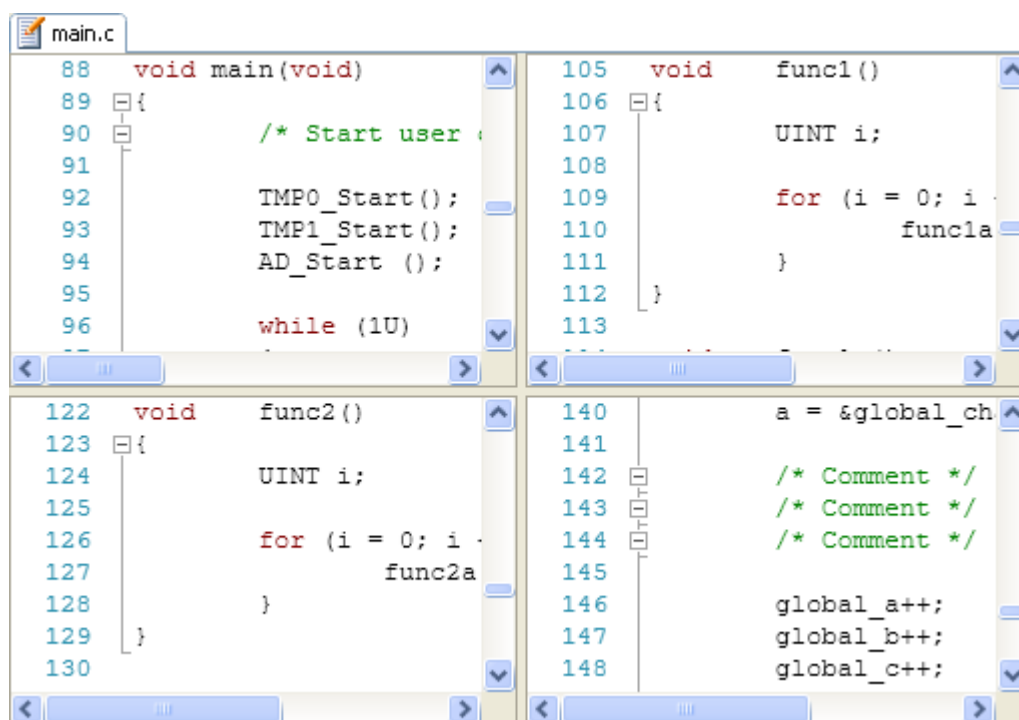
Save file name	Saves the contents of the opened text file.
Copy Full Path	Copies the full path of the opened text file to the clipboard.
Open Containing Folder	Opens the folder where the text file is saved in Explorer.

(2) Splitter bars

You can split the Editor panel by using the horizontal and vertical splitter bars within the view. This panel can be split up to four times.

- To split this panel, drag the splitter bar down or to the right to the desired position, or double-click any part of the splitter bar.
- To remove the split, double-click any part of the splitter bar.

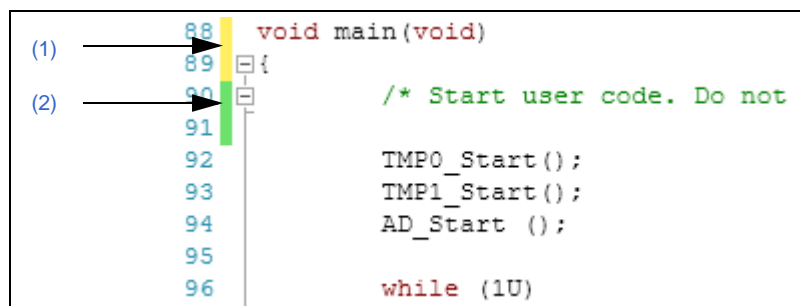
Figure A-22. Editor Panel (Four-way Split View)



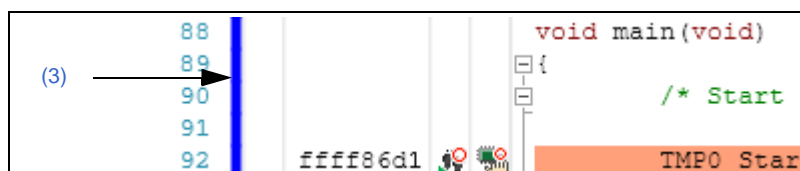
(3) Line number area

This area displays the line number of the opened text file or source file.

On each line there is an indicator that shows the line modification status.



This area changes color only when the downloaded module is out of date (using the "Warning" color of the [\[General - Font and Color\] category](#) of the [Option dialog box](#)). This area is valid only when connected to the debug tool and the downloaded source file is opened.



(1)		This means new or modified line but unsaved.
(2)		This means new or modified line and saved.
(3)		This means that the downloaded module is out of date.

(4) Coverage area

This area is valid only when connected to the debug tool and the downloaded source file is opened.

When the coverage function is valid, lines corresponding to the specified coverage measurement area are shown highlighted based on the code coverage measurement result that is acquired by executing the program (the color depends on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#)).

See "2.15 Measure Coverage [Simulator]" for details on the coverage measurement.

Hovering the mouse cursor over this area displays the area title "Coverage".

This area is provided with the following function.

(a) Clearing the coverage information via the menu

When you right-click in this area, the following menu below is displayed:

Clear Coverage Information	Clears all the coverage measurement results currently being stored in the debug tool. Note that this item appears only when the debug tool used supports the coverage function.
----------------------------	--

(5) Address area

This area is valid only when connected to the debug tool and the downloaded source file is opened.

This area shows the address corresponding to where the function is located in the memory space of the selected microcontroller.

The format of this area is fixed as hexadecimal number notation.

The address width corresponds to the one in memory space of the specified microcontroller in the project.

(6) Event area

This area is valid only when connected to the debug tool and the downloaded source file is opened.

Trace or timer events can be set at lines that have valid addresses.

In addition, the [Event mark](#) corresponding to an event that has been currently set is displayed.

Hovering the mouse cursor over this area (but not over a specific event mark) displays the area title "Event".

This area is provided with the following functions.

(a) Configuring events via the menu

When you right-click in this area, the following menu below is displayed:

Set Timer Start Event [E1] [E20]	Sets a Start Timer event to start measuring the execution time of the program when the line at caret is executed (see " 2.14.3 Measuring execution time in a section [E1] [E20] ").
Set Timer End Event [E1] [E20]	Sets a Stop Timer event to stop measuring the execution time of the program when the line at caret is executed (see " 2.14.3 Measuring execution time in a section [E1] [E20] ").
Set Trace Start Event	Sets a Start Tracing event to start collecting the trace data when the line at the caret is executed (see " 2.13.3 Collecting an execution history in a section ").
Set Trace End Event	Sets a Stop Tracing event to stop collecting the trace data when the line at the caret is executed (see " 2.13.3 Collecting an execution history in a section ").
Register Action Event...	Opens the Action Events dialog box to set an action event to the corresponding address of the line at the caret position.

(b) Changing the status of events via the menu

The events status can be changed from the following menu displayed by right-clicking the event mark.

Enable Event(s)	Changes all events state to an Enabled state. When an event is enabled and its condition is met, the event occurs.
Disable Event(s)	Changes all events state to a Disabled state. When an event is disabled and its condition is met, the event will not occur.
Delete Event(s)	Deletes all events.
View Details in Event Panel	Opens the Events panel to display the detailed information of the selected event.

(c) Pop-up display

By hovering the mouse cursor over the [Event mark](#), the name of the event, the detailed information for the event and the comments added to the event are a pop-up displayed.

When multiple events have been set in the applicable place, information for each event, up to a maximum of three events, is listed and displayed.


Remark The detailed information about the set event is reflected in the [Events panel](#).

(7) Main area

This area is valid only when the user is connected to the debug tool and the downloaded source file is opened.

Breakpoints can be set at lines that have valid addresses.

In addition, the [Event mark](#) corresponding to a breakpoint that has been currently set is displayed.

The current PC mark () that corresponds to the current PC position (PC register value) is displayed.

Note that the current PC mark is only displayed if the current PC value corresponds to the source text line, when the current PC position is modified or the state of the debug tool is changed from execution to stop.

Hovering the mouse cursor over this area (but not over a specific event mark) will display the area title "Main".

This area is provided with the following functions.

(a) Setting/deleting breakpoints

By clicking where you want to set a breakpoint with mouse, the breakpoints can be set easily.

The breakpoint is set to the instruction at the start address corresponding to the clicked line.

Once a breakpoint is set, an [Event mark](#) is displayed at the line that is set. In addition, the detailed information about the set breakpoint is reflected in the [Events panel](#).

When this operation is performed at a place where any one of the event marks is already being displayed, that event is deleted and the setting of breakpoints cannot be done.

See "[2.10.2 Stop the program at the arbitrary position \(breakpoint\)](#)" for details on how to set the breakpoint.



(b) Configuring breakpoints via the menu


When you right-click in this area, the following menu below is displayed:

Set Breakpoint	Sets a breakpoint in the area at the current source line (see " 2.10.2 Stop the program at the arbitrary position (breakpoint) "). [E1] [E20] By default the debug tool will set a hardware break when resources are available. This behavior can be customized by using the " Hardware Break First " or " Software Break First " menu items.
Set Hardware Breakpoint [E1] [E20]	Sets a breakpoint (Hardware Break event) to the line at the caret position.
Set Software Breakpoint [E1] [E20]	Sets a breakpoint (Software Break event) to the line at the caret position.
Hardware Break First [E1] [E20]	The type of break that can be set by a one click operation of the mouse is set as a hardware breakpoint (this is reflected in the setting of the [First using type of breakpoint] property in the [Break] [E1] [E20] category from the [Debug Tool Settings] tab on the Property panel).
Software Break First [E1] [E20]	The type of break that can be set by a one click operation of the mouse is set as a software breakpoint (this is reflected in the setting of the [First using type of breakpoint] property in the [Break] [E1] [E20] category from the [Debug Tool Settings] tab on the Property panel).

(c) Changing the status of breakpoints via the menu

The events status can be changed from the following menu displayed by right-clicking the event mark.

Enable Breakpoint	Changes the selected breakpoint state to an Enabled state. Event occurs when the specified condition is met. When the event mark () which indicates that multiple events have been set is selected, all of the breakpoints that have been set are enabled.
Disable Breakpoint	Changes the selected breakpoint state to a Disabled state. Event does not occur when the specified condition is met. When the event mark () which indicates that multiple events have been set is selected, all of the breakpoints that have been set are disabled.

Delete Breakpoint	Deletes the selected breakpoint. When the event mark () which indicates that multiple events have been set is selected, all of the breakpoints that have been set are deleted.
View Details in Event Panel	Opens the Events panel to display the detailed information of the selected event.

(d) Pop-up display

By hovering the mouse cursor over the [Event mark](#), the name of the event, the detailed information for the event and the comments added to the event are a pop-up displayed.

When multiple events have been set in the applicable place, information for each event, up to a maximum of three events, is listed and displayed.

Remark The detailed information about the set event is reflected in the [Events panel](#).

(8) Characters area

This area displays character strings of text files and source files and you can edit it.







This area has the following functions.

(a) Code outlining

This allows you to expand and collapse source code blocks so that you can concentrate on the areas of code which you are currently modifying or debugging. This is only available for only C and C++ source file types.

This is achieved by clicking the plus and minus symbols to the left of the Characters area.

Types of source code blocks that can be expanded or collapsed are:

Open and close braces ('{' and '}')	 { ... }
Multi-line comments ('/*' and '*/')	 /* */
Pre-processor statements ('if', 'elif', 'else', 'endif')	 #if[Preprocessor block]  #elif[Preprocessor block]  #else[Preprocessor block]  #endif

Caution This will be disabled for source files larger than 1MB.

(b) Characters editing

Characters can be entered from the keyboard.

Various shortcut keys can be used to enhance the edit function.

(c) Tag jump

If the information of a file name, a line number and a column number exists in the line at the caret position, selecting [Tag Jump] from the context menu opens the file in the Editor panel and jumps to the corresponding line and the corresponding column (if the target file is already opened in the Editor panel, you can jump to the panel).

See "[Table 2-4. Operation of Tag Jump](#)" for details on the operation of the tag jump.

(d) Current PC line display

When the current PC position (PC register value) corresponds to the source text lines, those lines are shown highlighted (the highlighting color depends on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#)).

This function is only enabled when connected to the debug tool and the source file is opened.

(e) Lines with breakpoints display

Lines where the breakpoints are set are shown highlighted (the highlighting color depends on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#)).

This function is only enabled when connected to the debug tool and the source file is opened.

(f) Code coverage measurement result display [Simulator]

When the coverage function is valid, lines corresponding to the specified coverage measurement area are shown highlighted based on the code coverage measurement result that is acquired by executing the program (the highlighting color depends on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#)).

See "[2.15 Measure Coverage \[Simulator\]](#)" for details on the coverage measurement.

This function is only enabled when connected to the debug tool and the source file is opened.

(g) Pop-up display of variables

When hovering the mouse cursor over a variable in the source text, a pop-up that shows the name and value of the variable is displayed ("*<variable name>=<variable value>*").

The display format of the variable value is same as "[Table A-10. Display Format of Watch Expressions \(Default\)](#)" depending on the type of the variable.

This function is only enabled when connected to the debug tool and the source file is opened.

(h) Setting of various events

Various events can be set to the addresses or lines where the caret currently exists by selecting [Break Settings], [Trace Settings] or [Timer Settings] from the context menu.

The corresponding [Event mark](#) is displayed in the [Event area](#) by setting the event. In addition, the detailed information about the set event is reflected in the [Events panel](#).

This function is only enabled when connected to the debug tool and the source file is opened.

See the following for details on how to set events.

- "[2.10.3 Stop the program at the arbitrary position \(break event\) \[E1\] \[E20\]](#)"
- "[2.10.4 Stop the program with the access to variables/I/O registers](#)"
- "[2.13.3 Collecting an execution history in a section](#)"
- "[2.13.4 Collecting an execution history only when conditions are met](#)"
- "[2.14.3 Measuring execution time in a section \[E1\] \[E20\]](#)"

Remark A breakpoint can be set or deleted easily in the [Main area](#) as well (see "[\(a\) Setting/deleting breakpoints](#)").

(i) Registering watch-expression

Variable names of C language, CPU registers, I/O registers, and assembler symbols can be registered in the [Watch panel](#) as watch-expressions.

See "[\(1\) Registering watch expressions](#)" for details on how to operate it.

This function is only enabled when connected to the debug tool and the source file is opened.

(j) File monitor

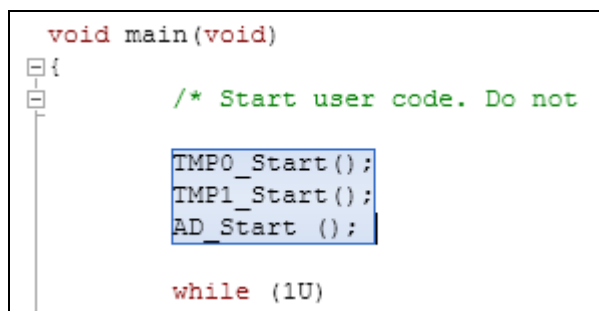
The following function for monitoring is provided to manage source files.

- A message is displayed when the downloaded load module file is older than the source file to open.
- If the contents of the currently displayed file have been changed without using CubeSuite+, a message will appear asking you whether you wish to save the file or not.

(k) Selecting blocks

You can select a block that consists of multiple lines by using the [Alt] key + left-mouse button combination.

- To select a block, press the [Alt] key and drag the left-mouse button.



Editing of the selected block can be done by using [Cut], [Copy], [Paste], or [Delete] in the [Edit] menu.

(l) Zoom in or out on a view

You can zoom in and out of the editor view by using the [Ctrl] key + mouse-wheel combination.

- Using the [Ctrl] key + mouse-wheel forward will zoom into the view, making the contents larger and easier to see (max. 300%).
- Using the [Ctrl] key + mouse-wheel backward will zoom out of the view, making the contents smaller (min. 50%).

Remark The following items can be customized by setting the [Option dialog box](#).

- Display fonts
- Tab interval
- Show or hide whitespace marks
- Colors of reserved words/comments

[[File] menu (Editor panel-dedicated items)]

The following items are exclusive for the [File] menu in the Editor panel (other items are common to all the panels).

Close <i>file name</i>	Closes the currently editing Editor panel. When the contents of the panel have not been saved, a confirmation message is shown.
Save <i>file name</i>	Overwrites the contents of the currently editing Editor panel. Note that when the file has never been saved or the file is read only, the same operation is applied as the selection in [Save <i>file name</i> As...].
Save <i>file name</i> As...	Opens the Save As dialog box to newly save the contents of the currently editing Editor panel.
<i>File name</i> Save Settings...	Opens the Save Settings dialog box to change the encoding and newline code of the file being opened in the currently editing Editor panel.
Print...	Opens the Windows dialog box for printing the contents of the currently editing Editor panel.
Print Preview	Opens the Print Preview window to preview the file contents to be printed.

[[Edit] menu (Editor panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Editor panel (other items are all invalid).

Undo	Cancels the previous operation and restores the characters and the caret position (up to 100 times).
Redo	Cancels the previous [Undo] operation and restores the characters and the caret position.
Cut	Cuts the selected character string and copies it to the clipboard. If there is no selection, the entire line is cut.
Copy	Copies the contents of the selected range to the clipboard as character string(s). If there is no selection, the entire line is copied.
Paste	Inserts (insert mode) or overwrites (overwrite mode) the characters that are copied on the clipboard into the caret position. When the contents of the clipboard are not recognized as characters, the operation is invalid. The mode selected for the current source file is displayed on the status bar (see "(b) Focus panel status information").
Delete	Deletes one character at the caret position. When there is a selection area, all the characters in the area are deleted.
Select All	Selects all the characters from beginning to the end in the currently editing text file.
Find...	Opens the Find and Replace dialog box, with its [Quick Find] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Quick Replace] tab selected.
Go To...	Opens the Go to Line dialog box to move the caret to the specified line.
Outlining	Displays a cascading menu for controlling expand and collapse states of source file outlining (see "(a) Code outlining").
Collapse to Definitions	Collapses all nodes that are marked as implementation blocks (e.g. function definitions).
Toggle Outlining Expansion	Toggles the current state of the innermost outlining section in which the cursor lies when you are in a nested collapsed section.
Toggle All Outlining	Toggles the collapsed state of all outlining nodes, setting them all to the same expanded or collapsed state. If there is a mixture of collapsed and expanded nodes, all nodes will be expanded.
Stop Outlining	Stops code outlining and remove all outlining information from source files.
Start Automatic Outlining	Starts automatic code outlining and automatically displayed in supported source files.
Advanced	Displays a cascading menu for performing an advanced operation for the Editor panel.
Increase Line Indent	Increases the indentation of the current cursor line by one tab.
Decrease Line Indent	Decreases the indentation of the current cursor line by one tab.
Uncomment Lines	Removes the first set of line-comment delimiters from the start of the current cursor line, appropriate to the current language (e.g. C++). This operation will only be available when the language of the current source file has line-comment delimiters specified (e.g. C++).
Comment Lines	Places line-comment delimiters at the start of the current cursor line, appropriate to the current language (e.g. C++). This operation will only be available when the language of the current source file has line-comment delimiters specified (e.g. C++).
Convert Tabs to Spaces	Converts all tabs on the current cursor line into spaces.

Convert Spaces to Tabs	Converts each set of consecutive space characters on the current line to tab characters, but only for those sets of spaces that are at least equal to one tab size.
Tabify Selected Lines	Tabifies the current line, causing all spaces at the start of the line (prior to any text) to be converted to tabs where possible.
Untabify Selected Lines	Untabifies the current line, causing all tabs at the start of the line (prior to any text) to be converted to spaces.
Make Uppercase	Converts all letters within the selection to uppercase.
Make Lowercase	Converts all letters within the selection to lowercase.
Toggle Character Casing	Toggles the character cases (uppercase / lowercase) of all letters within the selection.
Capitalize	Capitalizes the first character of every word within the selection.
Delete Horizontal Whitespace	Deletes any excess white space either side of the cursor position, leaving only one whitespace character remaining. If there the cursor is within a word or not surrounded by whitespace, this operation will have no effect.
Trim Trailing Whitespace	Deletes any trailing whitespace that appears after the last non-whitespace character on the cursor line.
Delete Line	Completely delete the current cursor line.
Duplicate Line	Duplicates the cursor line, inserting a copy of the line immediately after the cursor line.
Delete Blank Lines	Deletes the line at the cursor if it is empty or contains only whitespace.

[[Window] menu (Editor panel-dedicated items)]

The following items are exclusive for the [Window] menu in the Editor panel (other items are common to all the panels).

Split	Splits the active Editor panel horizontally. Only the active Editor panel can be split. Other panels will not be split. A panel can be split up to two times.
Remove Split	Removes the split view of the Editor panel.

[Context menu]

[Characters area/Line number area (while disconnecting from the debug tool)]

Cut	Cuts the selected character string and copies it to the clipboard. If there is no selection, the entire line is cut.
Copy	Copies the contents of the selected range to the clipboard as character string(s). If there is no selection, the entire line is copied.
Paste	Inserts (insert mode) or overwrites (overwrite mode) the characters that are copied on the clipboard into the caret position. When the contents of the clipboard are not recognized as characters, the operation is invalid. The mode selected for the current source file is displayed on the status bar (see "(b) Focus panel status information").
Find...	Opens the Find and Replace dialog box, with its [Quick Find] tab selected.
Go To...	Opens the Go to Line dialog box to move the caret to the specified line.
Jump to Function	Jumps to the function that is selected or at the caret position regarding the selected characters and the words at the caret position as functions (see "(4) Jump to functions").

Tag Jump	Jumps to the corresponding line and column in the corresponding file if the information of a file name, a line number and a column number exists in the line at the caret position (see "(c) Tag jump").
Advanced	Displays a cascading menu for performing an advanced operation for the Editor panel.
Increase Line Indent	Increases the indentation of the current cursor line by one tab.
Decrease Line Indent	Decreases the indentation of the current cursor line by one tab.
Uncomment Lines	Removes the first set of line-comment delimiters from the start of the current cursor line, appropriate to the current language (e.g. C++). This operation will only be available when the language of the current source file has line-comment delimiters specified (e.g. C++).
Comment Lines	Places line-comment delimiters at the start of the current cursor line, appropriate to the current language (e.g. C++). This operation will only be available when the language of the current source file has line-comment delimiters specified (e.g. C++).
Convert Tabs to Spaces	Converts all tabs on the current cursor line into spaces.
Convert Spaces to Tabs	Converts each set of consecutive space characters on the current line to tab characters, but only for those sets of spaces that are at least equal to one tab size.
Tabify Selected Lines	Tabifies the current line, causing all spaces at the start of the line (prior to any text) to be converted to tabs where possible.
Untabify Selected Lines	Untabifies the current line, causing all tabs at the start of the line (prior to any text) to be converted to spaces.
Make Uppercase	Converts all letters within the selection to uppercase.
Make Lowercase	Converts all letters within the selection to lowercase.
Toggle Character Casing	Toggles the character cases (uppercase / lowercase) of all letters within the selection.
Capitalize	Capitalizes the first character of every word within the selection.
Delete Horizontal Whitespace	Deletes any excess white space either side of the cursor position, leaving only one whitespace character remaining. If there the cursor is within a word or not surrounded by whitespace, this operation will have no effect.
Trim Trailing Whitespace	Deletes any trailing whitespace that appears after the last non-whitespace character on the cursor line.
Delete Line	Completely delete the current cursor line.
Duplicate Line	Duplicates the cursor line, inserting a copy of the line immediately after the cursor line.
Delete Blank Lines	Deletes the line at the cursor if it is empty or contains only whitespace.

[Characters area/Line number area (while connecting to the debug tool)]

Register to Watch1	Registers a selected character string or a word at the caret position to the Watch panel (Watch1) as a watch-expression (the judgment of the word depends on current build tool). Note that this item becomes invalid when no corresponding address exists in the line at caret.
Register Action Event...	Opens the Action Events dialog box to set an action event to the corresponding address of the line at the caret position ^{Note 1} . Note that this item becomes invalid when no corresponding address exists in the line at caret.
Cut	Cuts the selected character string and copies it to the clipboard. If there is no selection, the entire line is cut.

Copy	Copies the contents of the selected range to the clipboard as character string(s). If there is no selection, the entire line is copied.
Paste	Inserts (insert mode) or overwrites (overwrite mode) the characters that are copied on the clipboard into the caret position. When the contents of the clipboard are not recognized as characters, the operation is invalid. The mode selected for the current source file is displayed on the status bar (see "(b) Focus panel status information").
Find...	Opens the Find and Replace dialog box, with its [Quick Find] tab selected.
Go To...	Opens the Go to Line dialog box to move the caret to the specified line.
Forward To Next Cursor Position	Forwards to the position before operating [Back To Last Cursor Position].
Back To Last Cursor Position	Goes back to the position before operating [Jump to Function].
Go to Here	Executes the program from the address indicated by the current PC value to the address corresponding to the line at the caret position ^{Note 1} . If the corresponding address of the line at the caret position does not exist, the program is executed to the corresponding address of the lower valid line. Note that this item becomes invalid during execution of a program/[Build & Download].
Set PC to Here	Sets the address of the line at the current caret position to the current PC value ^{Note 1} . Note that this item becomes invalid when no corresponding address exists in the line at caret, or during execution of a program/[Build & Download].
Jump to Function	Jumps to the function that is selected or at the caret position regarding the selected characters and the words at the caret position as functions (see "(4) Jump to functions").
Tag Jump	Jumps to the corresponding line and column in the corresponding file if the information of a file name, a line number and a column number exists in the line at the caret position (see "(c) Tag jump").
Jump to Disassemble	Opens the Disassemble panel and jumps to the address corresponding to the line at the caret ^{Note 1} . Note that this item becomes invalid when no corresponding address exists in the line at caret.
Advanced	Displays a cascading menu for performing an advanced operation for the Editor panel.
Increase Line Indent	Increases the indentation of the current cursor line by one tab.
Decrease Line Indent	Decreases the indentation of the current cursor line by one tab.
Uncomment Lines	Removes the first set of line-comment delimiters from the start of the current cursor line, appropriate to the current language (e.g. C++). This operation will only be available when the language of the current source file has line-comment delimiters specified (e.g. C++).
Comment Lines	Places line-comment delimiters at the start of the current cursor line, appropriate to the current language (e.g. C++). This operation will only be available when the language of the current source file has line-comment delimiters specified (e.g. C++).
Convert Tabs to Spaces	Converts all tabs on the current cursor line into spaces.
Convert Spaces to Tabs	Converts each set of consecutive space characters on the current line to tab characters, but only for those sets of spaces that are at least equal to one tab size.
Tabify Selected Lines	Tabifies the current line, causing all spaces at the start of the line (prior to any text) to be converted to tabs where possible.
Untabify Selected Lines	Untabifies the current line, causing all tabs at the start of the line (prior to any text) to be converted to spaces.

Make Uppercase	Converts all letters within the selection to uppercase.
Make Lowercase	Converts all letters within the selection to lowercase.
Toggle Character Casing	Toggles the character cases (uppercase / lowercase) of all letters within the selection.
Capitalize	Capitalizes the first character of every word within the selection.
Delete Horizontal Whitespace	Deletes any excess white space either side of the cursor position, leaving only one whitespace character remaining. If there the cursor is within a word or not surrounded by whitespace, this operation will have no effect.
Trim Trailing Whitespace	Deletes any trailing whitespace that appears after the last non-whitespace character on the cursor line.
Delete Line	Completely delete the current cursor line.
Duplicate Line	Duplicates the cursor line, inserting a copy of the line immediately after the cursor line.
Delete Blank Lines	Deletes the line at the cursor if it is empty or contains only whitespace.
Break Settings	The following cascade menus are displayed to set the break-related event. Note that events can be set only for lines for which events can be set (see "(6) Event area").
Set Hardware Break	Sets a breakpoint (Hardware Break event) to the line at the caret position (see "2.10.2 Stop the program at the arbitrary position (breakpoint)") ^{Note 1} .
Set Software Break [E1] [E20]	Sets a breakpoint (Software Break event) to the line at the caret position (see "2.10.2 Stop the program at the arbitrary position (breakpoint)") ^{Note 1} .
Set Read Break to [Simulator]	Sets a break event with read access condition to the line at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "(1) Set a break event (access type) to a variable/I/O register").
Set Write Break to [Simulator]	Sets a break event with write access condition to the line at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "(1) Set a break event (access type) to a variable/I/O register").
Set R/W Break to [Simulator]	Sets a break event with read/write access condition to the line at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "(1) Set a break event (access type) to a variable/I/O register").
Set Combination Break [E1] [E20]	Sets a break at the caret position's address or the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break event (see "(1) Set a break event (execution type)").
Set Read Combination Break to [E1] [E20]	Sets a break event by a read access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break (see "(1) Set a break event (access type) to a variable/I/O register").
Set Write Combination Break to [E1] [E20]	Sets a break event by a write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break (see "(1) Set a break event (access type) to a variable/I/O register").
Set R/W Combination Break to [E1] [E20]	Sets a break event by a read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break (see "(1) Set a break event (access type) to a variable/I/O register").
Break Option	Opens the Property panel to set the break function.

Trace Settings	The following cascade menus are displayed to set the trace-related event. Note that events can be set only for lines for which events can be set (see "(6) Event area").
Start Tracing	Sets a Start Tracing event to start collecting the trace data when an instruction of an address at the caret position is executed (see "(1) Setting a trace start event and a trace end event"). [Simulator] In addition, the selecting of the [Use trace function] property in the [Trace] category from the [Debug Tool Settings] tab on the Property panel is automatically set to [Yes].
Stop Tracing	Sets a Stop Tracing event to stop collecting the trace data when an instruction of an address at the caret position is executed (see "(1) Setting a trace start event and a trace end event"). [Simulator] In addition, the selecting of the [Use trace function] property in the [Trace] category from the [Debug Tool Settings] tab on the Property panel is automatically set to [Yes].
Record Reading Value	Sets a Point Trace event to record the access value as the trace data when the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is read accessed (see "(1) When an access to a variable or I/O register occurred").
Record Writing Value	Sets a Point Trace event to record the access value as the trace data when the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is write accessed (see "(1) When an access to a variable or I/O register occurred").
Record R/W Value	Sets a Point Trace event to record the access value as the trace data when the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is read/ write accessed (see "(1) When an access to a variable or I/O register occurred").
Record Start R/W Value	Sets a trace event that causes trace recording to start upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(1) Setting a trace start event and a trace end event").
Record End R/W Value	Sets a trace event that causes trace recording to end upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(1) Setting a trace start event and a trace end event").
Show Trace Result	Opens the Trace panel and displays the acquired trace data.
Trace Settings	Opens the Property panel to set the trace function.
Timer Settings [E1] [E20]	The following cascade menus are displayed to set the timer-related event (see "2.14.3 Measuring execution time in a section [E1] [E20]"). Note that events can be set only for lines for which events can be set (see "(6) Event area").
Start timer	Sets a Start Timer event to start measuring the execution time of the program when the line at caret is executed ^{Note 1} (see "(a) How to set a timer start event").
Stop timer	Sets a Stop Timer event to stop measuring the execution time of the program when the line at caret is executed ^{Note 1} (see "(b) How to set a timer end event").
Set Timer Start R/W Value	Sets a timer start event that causes a measurement of the program's execution time to start upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(a) How to set a timer start event").
Set Timer <N>	Specify a channel ^{Note 2} in which a timer start event is set.
Set Timer End R/W Value	Sets a timer end event that causes a measurement of the program's execution time to finish upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(b) How to set a timer end event").

Set Timer </>	Specify a channel ^{Note 2} in which a timer start event is set.
View Result of Timer	Opens the Events panel and displays only timer-related events.
Clear Coverage Information [Simulator]	Clears all the coverage measurement results currently being stored in the debug tool. Note that this item appears only when the debug tool used supports the coverage function.

- Notes 1.** A message is displayed if these items are selected when the downloaded load module file is older than the opened source file.
- 2.** The specifiable number of channels differs between the RX600 and RX200 Series, as shown below.
RX600 Series; 2 (32 bits * 2) or 1 (64 bits * 1)
RX200 Series: 1 (24 bits * 1)

Memory panel

This panel displays or changes memory contents (see Section "2.11.1 Displaying and changing memory contents").

Up to four memory panels can be opened at a time, with each panel discriminated by the name in their title bar, "Memory 1," "Memory 2," "Memory 3," or "Memory 4."

When memory values change after execution of the program, the display is automatically updated. (During step execution, the display is successively updated each time a step is executed.)

Also, if the [Realtime display update function](#) is enabled, the display of values can be updated in real time, even while the program is under execution.

Note that this panel can only be opened when CubeSuite+ is connected with the debug tool.


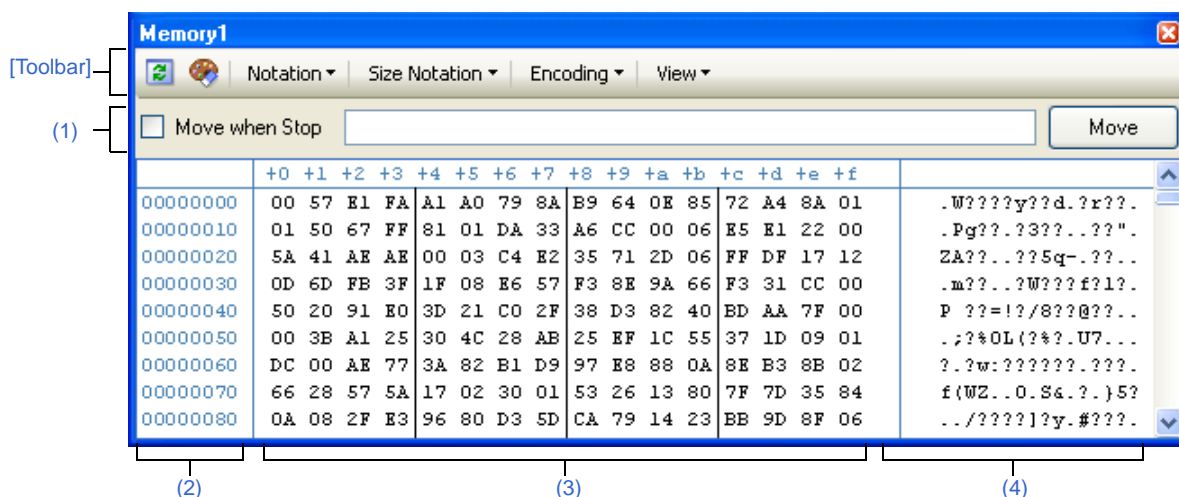
Remark You can set the scroll range of the vertical scroll bar of this panel in the [Scroll Range Settings dialog box](#) which opens by clicking the toolbar button .

Figure A-23. Memory Panel



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[\[File\] menu \(Memory panel-Only Items\)\]](#)
- [\[\[Edit\] menu \(Memory panel-Only Items\)\]](#)
- [\[Context Menu\]](#)

[How to open]

- Choose [Memory] from the [View] menu and then select [Memory 1-4].

[Description of each area]

(1) Display position specification area

By specifying an address expression, it is possible to specify the display start position of memory values.

Make the following specifications in order.

(a) Specify address expressions

Enter an address expression for the address of the memory value to be displayed directly in the text box. Input expressions in up to 1,024 characters each can be specified, with their calculation result handled as a display start position.

However, address expressions greater than the microcontroller's address space cannot be specified.

- Remarks 1.** By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 Symbol name completion function").
- 2.** In cases where the specified address expression represents a symbol and the size is known, a range of memory from the start address to end address of that symbol is selected when displayed.

(b) Specify whether to evaluate address expression manually or automatically

The [Move when stop] checkbox and the [Move] buttons are used to determine the timing with which the display start position is changed.

[Move when Stop]	<input checked="" type="checkbox"/>	The address expression is automatically evaluated after the program has halted and the caret moves to the address derived from that calculation.
	<input type="checkbox"/>	The address expression is not automatically evaluated after the program has halted. In this case, the address expression is evaluated by clicking the [Move] button.
[Move] button		If the [Move when Stop] checkbox is not checked, the address expression is evaluated by clicking this button and the caret moves to the address derived from that calculation.

(2) Address area

This area displays memory addresses (always in hexadecimal).

The address width matches that of the microcontroller's memory space specified in the project.

This area cannot be edited.


(3) Memory value area

This area displays or changes memory values for the program mapped to the microcontroller.

By default, the display starts from the address "0."

To specify the display notation and display width of memory values, use the toolbar buttons or select [Notation] and [Size Notation] from the context menu. (By default, memory values are displayed in hexadecimal notation and 8-bit width.)

The following table describes the meaning of marks and colors used to represent memory values (The colors in which text and backgrounds are displayed depend on how the [General - Font and Color] category of the Option dialog box is set.).

Example display (default)			Description
00	Text color	Blue	Memory values that have been changed by the user (Pressing the [Enter] key will write them into the target memory)
	Background color	Standard color	
<u>00</u> (Underlined)	Text color	Standard color	Memory values at addresses for which symbols are defined (Register watch expressions can be registered)
	Background color	Standard color	
00	Text color	Brown	Memory values that have changed as a result of program execution Note Clicking the  button in the toolbar resets the highlighting.
	Background color	Cream	

Example display (default)			Description		
00	Text color	Pink	Memory values for which the Realtime display update function is enabled		
	Background color	Standard color			
00	Text color	Standard color	Read/ Fetch	Current access condition of the memory value when the Realtime display update function is enabled	
	Background color	Palegreen			
00	Text color	Standard color	Write		
	Background color	Orange			
00	Text color	Standard color	Read and Write		
	Background color	Paleturquoise			
00	Text color	White	Lost		
	Background color	LightGray			
00	Text color	Gray	Memory values of not-readable area		
	Background color	Standard color			
??	Text color	Gray	Areas not memory-mapped or areas not rewritable (e.g., IO register and I/O protection areas) or when acquisition of memory values failed		
	Background color	Standard color			
**	Text color	Standard color	When areas other than the Realtime display update area are selected for display during program execution or when acquisition of memory values failed		
	Background color	Standard color			

Note Applicable only to the memory values in the address area that was displayed on the Memory panel immediately before the program execution. As the comparison is made before and after the program execution, highlighting will not be used if the value remained unchanged.

This area has the following features.

(a) Popup display

When the mouse cursor is placed on top of a memory value, the following content is displayed in a popup box, relative to the symbol nearest in forward direction to the address indicated by the mouse cursor. However, if symbol information is nonexistent (i.e., not underlined), no popup is displayed.

variable	+ 0x14
Symbol name	Offset value

Symbol name	A symbol name is displayed.
Offset value	If the address has no symbols defined, an offset from the symbol nearest in forward to it is displayed (always in hexadecimal).

(b) Realtime display update function

Using the realtime display update function, it is possible to display or change memory values even while the program is under execution, not just when the program is halted.

For details about the realtime display update function, see "(4) [Displaying and changing memory contents during program execution](#)"

(c) Edit memory values

You can edit a memory value directly from the keyboard by simply moving the caret to the memory value you wish to change.

When a memory value is edited, the altered part of it changes in display color. While in this state, hit the [Enter] key, and the changed value is written into the target memory (Pressing the [Esc] key before pressing [Enter] key cancels editing.).

For details on how to change memory values, see "(3) [Changing memory contents](#)".

(d) Search and initialize memory values

Open the [Memory Search dialog box](#) in which you can search memory contents in the specified address area by selecting [Find...] from the context menu.

Also, selecting [Fill...] from the context menu will open the [Memory Initialize dialog box](#) in which you can collectively modify memory contents in the specified address range.

(e) Copy and paste

By selecting a range of memory values with the mouse, you can copy the content of the selected part as a character string to the clipboard, and then paste it to the caret position.

To perform these operations, select the appropriate item from the context menu or from the [Edit] menu.

Note that pasting is possible only when the display format (notation and bit width) of the character string matches that of the area to which it is pasted (A message will appear if the formats do not agree).

The following table shows the character code and character strings that can be used in this area (A message will appear when a character string other than those listed here is pasted.).

Character code	ASCII
Character string	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, A, B, C, D, E, F

(f) Register watch expressions

A memory value at the addresses for which symbol is defined is underlined to indicate that it can be registered as a watch expression.

Select this underlined memory value, or while the caret is positioned at one of such memory values, select [Register to Watch1] from the context menu, and the symbol name at the specified address is registered as a watch expression on the [Watch panel](#) (Watch1).

Caution Memory values with no underlines cannot be registered as watch expressions.

(g) Save memory values

Choosing [Save Memory Data As...] from the [File] menu opens the [Data Save dialog box](#), allowing the contents of this panel to be saved to a text file (*.txt) or CSV file (*.csv).

For details on how to save memory values, see "(7) [Saving displayed memory contents](#)".

(4) Character string area

This area displays memory values converted into character code.

To specify character code, click the appropriate toolbar button or select [Encoding] from the context menu.

This area has the following features.

(a) Edit character strings

In the current version of the software, character strings can be changed only when [ASCII] is specified for the character code.

You can edit a character string directly from the keyboard by simply moving the caret to the character string you wish to change.

When a character string is edited, the altered part of it changes in display color. While in this state, hit the [Enter] key, and the changed value is written into the target memory (Pressing the [Esc] key before pressing [Enter] key cancels editing.).


















(b) Copying and pasting


By selecting a range of character strings with the mouse, you can copy it to the clipboard as character strings and then paste it to the caret position.

To perform these operations, select the appropriate item from the context menu or from the [Edit] menu.

Note that pasting is possible only when [ASCII] is specified for the character code (If any other character code is specified, a message is displayed.).

[Toolbar]

	Obtains latest information from the debug tool and updates display.
	Resets the highlighting that indicates the spot whose value has changed as a result of program execution. However, this button is disabled during program execution.
Notation	Shows the following buttons that change the form in which memory values are displayed. However, these buttons are disabled during program execution.
	Displays memory values in hexadecimal (default).
	Displays memory values in signed decimal.
	Displays memory values in unsigned decimal.
	Displays memory values in octal.
	Displays memory values in binary.
Size Notation	Shows the following buttons that change the form in which memory value size is displayed. However, these buttons are disabled during program execution.
	Displays memory values in 4-bit width.
	Displays memory values in 8-bit width (default).
	Displays memory values in 16-bit width. The value is converted according to the endian in the target memory area.
	Displays memory values in 32-bit width. The value is converted according to the endian in the target memory area.
	Displays memory values in 64-bit width. The value is converted according to the endian in the target memory area.
Encoding	Shows the following buttons that change the encode in which character strings are displayed. However, these buttons are disabled during program execution.
	Displays character strings in ASCII code (default).
	Displays character strings in Shift_JIS code.
	Displays character strings in EUC-JP code.
	Displays character strings in UTF-8 code.
	Displays character strings in UTF-16 code.
View	Shows the following buttons that change the display form.

	Opens the Scroll Range Settings dialog box to set the scroll range.
---	---

[[File] menu (Memory panel-Only Items)]

The [File] menu used exclusively for the memory panel is as follows. (The other items are shared.)

However, all of these items are disabled during program execution.

Save Memory Data	Saves memory contents to a text file (*.txt) or CSV file (*.csv) that has been saved previously (see "(g) Save memory values "). If this item is selected for the first time after startup, the same operation as [Save Memory Data As ...] will be performed.
Save Memory Data As...	Opens the Data Save dialog box in order to save memory to a specified text file (*.txt) or CSV file (*.csv) (see "(g) Save memory values ").

[[Edit] menu (Memory panel-Only Items)]

The [Edit] menu used exclusively for the memory panel is as follows. (All other items are disabled.)

However, all of these items are disabled during program execution.

Copy	Copies a selected range as character string to the clipboard.
Paste	Pastes the copied character string from the clipboard to the caret position. - To paste in the memory value area, see "(e) Copy and paste ". - To paste in the character string area, see "(b) Copying and pasting ".
Find...	Opens the Memory Search dialog box . A search is performed within the Memory value area or the Character string area whichever has the caret in it.

[Context Menu]

Register to Watch1	Registers the symbol at the caret position on the Watch panel (Watch1). When symbols are registered as watch expressions, they are registered as variable names. Because of this, displayed symbol names vary by scope. However, if the address corresponding to the memory value at the caret position has no symbols defined, this menu is disabled (see "(f) Register watch expressions ").
Find...	Opens the Memory Search dialog box . A search is performed within the Memory value area or Character string area whichever has the caret in it. However, this menu is disabled during program execution.
Fill...	Opens the Memory Initialize dialog box .
Refresh	Obtains latest information from the debug tool and updates the display.
Copy	Copies a selected range as character string to the clipboard. However, this menu is disabled during program execution.
Paste	Pastes the copied character string from the clipboard to the caret position. However, this menu is disabled during program execution. - To paste in the memory value area, see "(e) Copy and paste ". - To paste in the character string area, see "(b) Copying and pasting ".

Notation	Shows the following cascaded menu to specify the display notation in the memory value area. However, this menu is disabled during program execution.
Hexadecimal	Displays memory values in hexadecimal (default).
Signed Decimal	Displays memory values in signed decimal.
Unsigned Decimal	Displays memory values in unsigned decimal.
Octal	Displays memory values in octal.
Binary	Displays memory values in binary.
Size Notation	Shows the following cascaded menu to specify the display width in the memory value area. However, this menu is disabled during program execution.
4 Bits	Displays memory values in 4-bit width.
1 Byte	Displays memory values in 8-bit width (default).
2 Bytes	Displays memory values in 16-bit width. The value is converted according to the endian in the target memory area.
4 Bytes	Displays memory values in 32-bit width. The value is converted according to the endian in the target memory area.
8 Bytes	Displays memory values in 64-bit width. The value is converted according to the endian in the target memory area.
Encoding	Shows the following cascaded menu to specify the character code in the character string area. However, this menu is disabled during program execution.
ASCII	Displays character strings in ASCII code (default).
Shift_JIS	Displays character strings in Shift_JIS code.
EUC-JP	Displays character strings in EUC-JP code.
UTF-8	Displays character strings in UTF-8 code.
UTF-16	Displays character strings in UTF-16 code.
View	Shows the following cascaded menu to change the display form.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range.
Highlight Accessed	Checking this menu will highlight the memory value which has been changed due to program execution (default). However, this menu is disabled during program execution.
Periodic Updating	Shows the following cascaded menu to set realtime display updates (see "(b) Realtime display update function ").
Periodic Updating Options	Opens the Property panel to comprehensively set the Realtime display update.

Disassemble panel

This panel is used to display the results of disassembling the contents of the memory (disassembled text), and execute line assembly (see "2.6.4 Performing line assembly").

Furthermore, the instruction level debugging (see "2.9.3 Execute programs in steps") and the code coverage measurement result display **[Simulator]** (see "2.15 Measure Coverage [Simulator]") can be performed in this panel.

Up to a maximum of four of these panels can be opened. Each panel is identified by the names "Disassemble1", "Disassemble2", "Disassemble3" and "Disassemble4" on the titlebar.

The source text in the source file corresponding to the code data can also be displayed by setting to the mixed display mode (default).

This panel appears only when connected to the debug tool.

Caution A step execution is performed in instruction level units when the focus is in this panel (see "2.9.3 Execute programs in steps").


- Remarks 1.** Using the [Scroll Range Settings dialog box](#) that is opened by clicking the toolbar button , it is possible to set the range in which the vertical scroll bar of this panel can be scrolled.
- 2.** You can print the current screen image of this panel by selecting [Print...] from the [File] menu.

Figure A-24. Disassemble Panel (When Mixed Display Mode Is Selected)

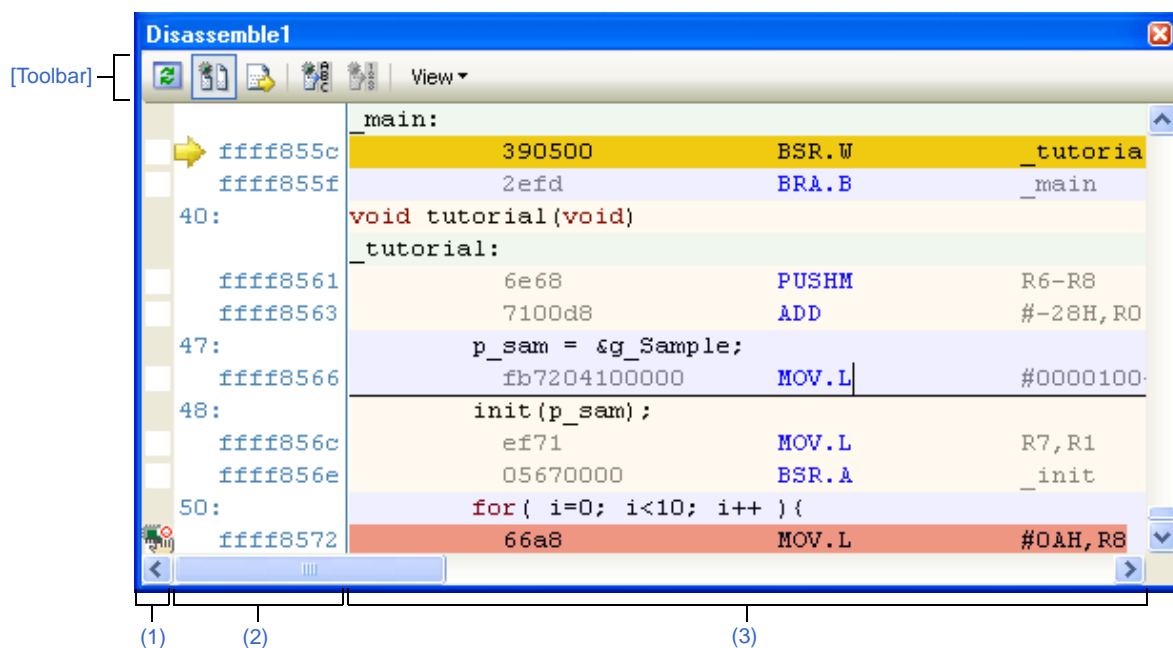


Figure A-25. Disassemble Panel (When Mixed Display Mode Is Not Selected)

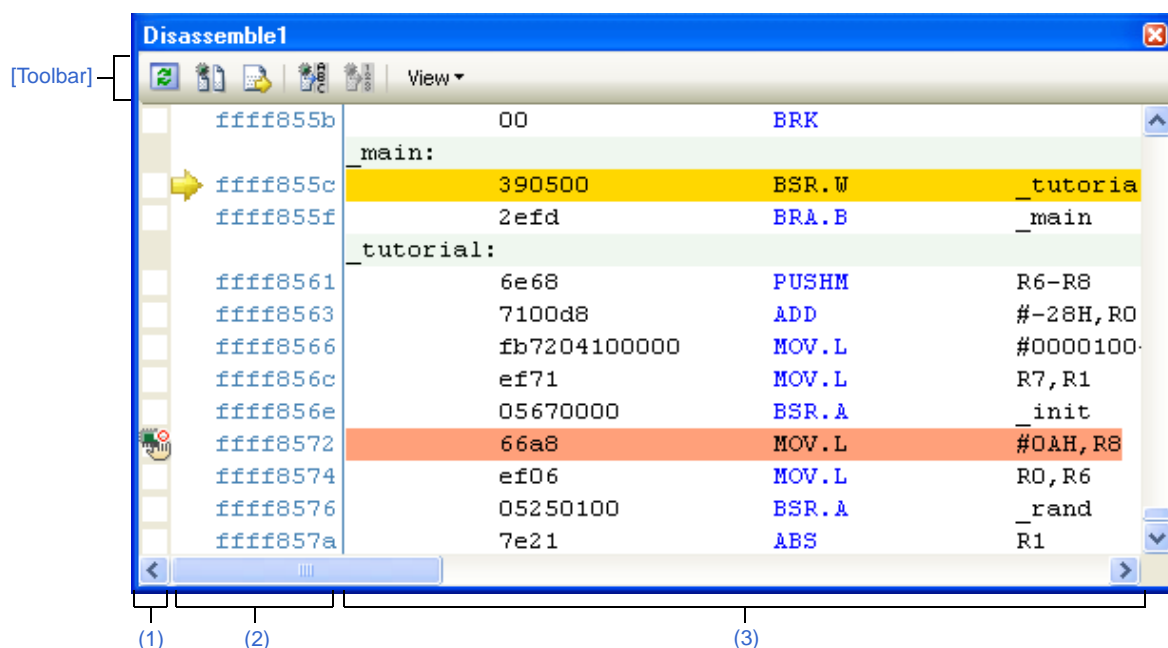
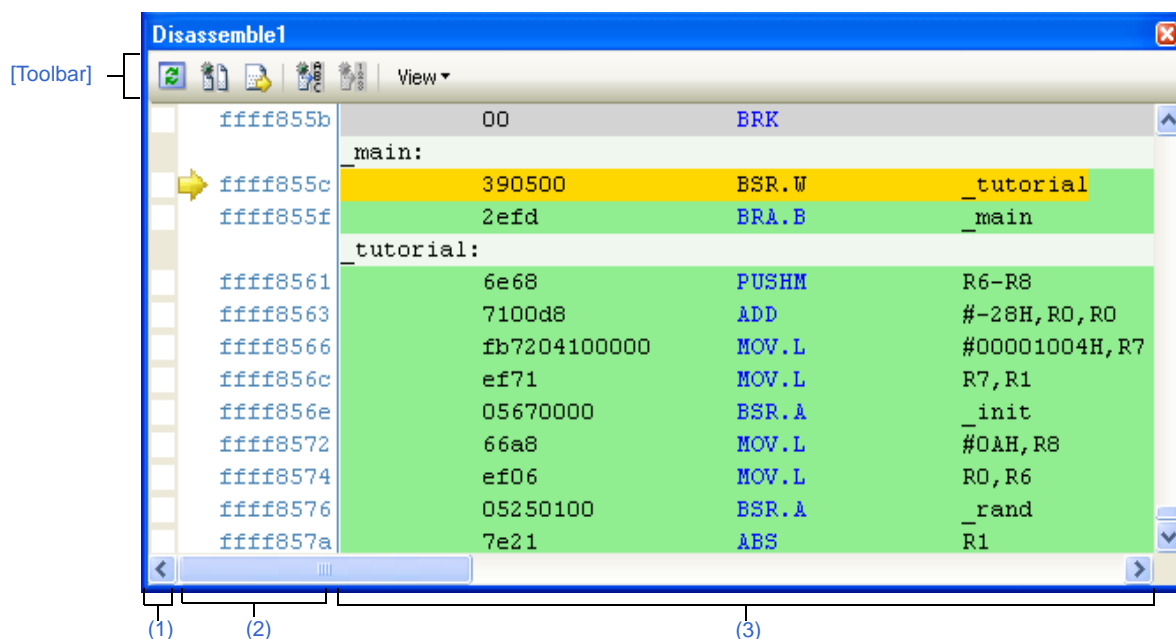


Figure A-26. Disassemble Panel (When Code Coverage Measurement Result Is Displayed) [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Disassemble panel-dedicated items)]
- [[Edit] menu (Disassemble panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Disassemble] >> [Disassemble 1 - 4].

[Description of each area]**(1) Event area**

The lines for which events can be set are shown with the background color in white (this mean that events cannot be set for those lines whose background color in gray).

In addition, the [Event mark](#) corresponding to an event that has been currently set is displayed.

This area is provided with the following functions.

(a) Setting/deleting breakpoints

By clicking where you want to set a breakpoint with the mouse, the breakpoint can be set easily.

The breakpoint is set to the instruction at the start address of the clicked line.

Once the breakpoint is set, the [Event mark](#) is displayed at the line that is set. In addition, the detailed information about the set breakpoint is reflected in the [Events panel](#).




When this operation is performed at a place where any one of the event marks is already being displayed, that event is deleted and the setting of breakpoints cannot be done.

Note that the setting of events can be done only for those lines where the background color is shown in white.

See "[2.10.2 Stop the program at the arbitrary position \(breakpoint\)](#)" for details on how to set the breakpoint.

(b) Changes event status

Event status can be changed from the following menu displayed by right-clicking the event mark.

Enable Event	Changes the selected event state to an Enabled state. Event occurs when the specified condition is met. When the event mark () which indicates that multiple events have been set is selected, all of the events that have been set are enabled.
Disable Event	Changes the selected event state to a Disabled state. Event does not occur when the specified condition is met. When the event mark () which indicates that multiple events have been set is selected, all of the events that have been set are disabled.
Delete Event	Deletes the selected event. When the event mark () which indicates that multiple events have been set is selected, all of the events that have been set are deleted.
View Event Detailed Setup	Opens the Events panel to display the detailed information of the selected event.


(c) Pop-up display

By superimposing the mouse cursor on the [Event mark](#), the name of the event, the detailed information for the event and the comments added to the event are pop-up displayed.

When multiple events have been set in the applicable place, information for each event, up to a maximum of three events, is listed and displayed.

(2) Address area

The address per line to start disassembling is displayed (hexadecimal number notation fixing).

In addition, the current PC mark () that corresponds to the current PC position (PC register value) is displayed.

The address width corresponds to the one in memory space of the specified microcontroller in the project.

For the source text line in the mixed display mode, line numbers (xxx:) in the source file correspond to the start address are displayed.

This area is provided with the following functions.

(a) Pop-up display

By superimposing the mouse cursor on a address or line number, the following information is pop-up displayed.

Address	Format: <Load module name> ^{Note 5} \$<Label name> + <Offset value> Example1: test1.out\$main + 0x10 Example2: sub function + 0x20
Source line number	Format: <Load module name> ^{Note 5} \$<File name> # <Line number> Example1: test1.out\$main.c#40 Example2: main.c#100

Note <Load module name> is displayed only when multiple load modules have been downloaded to the debug tool.



(3) Disassemble area

The results of disassembling are displayed next to the corresponding source text as follows.

Figure A-27. Display Contents of the Disassemble Area (In the Case of the Mixed Display Mode)

Label line	void change (long *a)
PC line	_change:
Breakpoint line	7100d8 ADD #-28H, R0, R0
Source text line	for (i=0; i<10; i++) {
Disassemble results	+3 66a2 MOV.L #0AH, R2
	+5 ef03 MOV.L R0, R3
	+7 ef14 MOV.L R1, R4
	tmp[i] = a[i];
	+9 fd2a45 MOV.L [R4+], R5
	+c fd2235 MOV.L R5, [R3+]
	+f 6012 SUB #1H, R2
	Offset value Code Instruction

Label line	The label is displayed when a label is defined for the address, and its corresponding line is shown highlighted in lightgreen.	
PC line	A line corresponding to an address of the current PC (PC register value) is shown highlighted ^{Note 1} .	
Breakpoint line	A line at which a breakpoint is set is shown highlighted ^{Note 1} .	
Source text line	The source text corresponding to the code data is displayed ^{Note 2} .	
Disassemble results	Offset value	The offset value from the nearest label is displayed when a label is defined for the address ^{Note 3} .
	Code	The code that is the target of disassembly is displayed in hexadecimal number.
	Instruction	Instruction is displayed as the result of disassembling. The mnemonics are shown highlighted in blue.

- Notes**
1. The highlighting color depends on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#).
 2. The source text can be set to non-display by clicking the  button (toggle) on the toolbar or removing the check for [Mixed Display] from the context menu (this option is checked by default).
 3. Offset values are not displayed by default. They can be displayed by clicking the  button on the toolbar or selecting [Show Offset] from the context menu.

This area is provided with the following functions.

(a) Line assembly

Instructions and code displayed in this panel can be edited (line assembly).

See ["2.6.4 Performing line assembly"](#) for details on how to operate it.

(b) Program execution by instruction level

Execution can be controlled at the instruction level unit by step executing a program in a state where there is a focus on this panel.

See ["2.9.3 Execute programs in steps"](#) for details on how to operate it.

(c) Setting of various events

Various events can be set to the addresses/lines where the caret currently exists by selecting [Break Settings], [Trace Settings] or [Timer Settings] from the context menu.

The corresponding [Event mark](#) is displayed in the [Event area](#) when an event is set. In addition, the detailed information about the set event is reflected in the [Events panel](#).

+Note, however, that the setting of events can be done only for those lines where the background color is shown in white in the event area.

See the following for details on how to set events.

- ["2.10.3 Stop the program at the arbitrary position \(break event\) \[E1\] \[E20\]"](#)
- ["2.10.4 Stop the program with the access to variables/I/O registers"](#)
- ["2.13.3 Collecting an execution history in a section"](#)
- ["2.13.4 Collecting an execution history only when conditions are met"](#)
- ["2.14.3 Measuring execution time in a section \[E1\] \[E20\]"](#)


Remark A breakpoint can be set or deleted easily in the [Event area](#) as well (see ["\(a\) Setting/deleting breakpoints"](#)).


(d) Registering watch-expression

Variable names of C language, CPU registers, I/O registers, and assembler symbols can be registered in the [Watch panel](#) as watch-expressions.

See ["\(1\) Registering watch expressions"](#) for details on how to operate it.

(e) Moving to symbol definition place

By clicking the  button on the toolbar or selecting [Go to Symbol] from the context menu in a state where the caret has been moved to a instruction that has referenced a symbol, the caret position is moved to the address where the symbol at the caret position has been defined.

In addition, when following on this operation you click on the  button on the toolbar or select [Back to Address] from the context menu, the caret position is returned to the instruction that has referenced a symbol before the caret was moved (the address value of the instruction that has referenced a symbol is displayed in *Address*).

(f) Jump to source line and memory

By selecting [Jump to Source] from the context menu, the [Editor panel](#) is opened with moving the caret to the source line corresponding to the address at the current caret position (if the Editor panel is already opened, jump to the panel).

In addition, by similarly selecting [Jump to Memory], the [Memory panel](#) (Memory1) is opened with moving the caret to the memory value corresponding to the address at the current caret position (if the Memory panel (Memory1) is already opened, jump to the panel).

(g) Code coverage measurement result display [Simulator]










When the coverage function is valid, lines corresponding to the specified coverage measurement area are shown highlighted based on the code coverage measurement result that is acquired by executing the program. See "[2.15 Measure Coverage \[Simulator\]](#)" for details on the coverage measurement.

(h) Saving the contents of disassembled data

The [Data Save dialog box](#) can be opened by selecting the [File] menu >> [Save Disassemble Data As...], and the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).

See "[\(5\) Saving the displayed contents of disassembled results](#)" for details on the method for saving the contents of disassembled data.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel.
	Sets to the mixed display mode and displays the correspondence between the disassembled data and the source text (default).
	Specifies the caret position so that it follows the current PC value.
	Moves the caret to the define position of the selected symbol.
	Moves the caret to the position (<i>address</i>) immediately before it is moved with the  button.
View	The following buttons to set the display contents in the disassemble area are displayed.
	Displays the offset value of the label. The offset value from the nearest label is displayed when a label is defined for the address.
	Displays the address value in the format "symbol + offset value" (default). Note that when a symbol has been defined as the address value, only the symbol is displayed.
	Opens the Scroll Range Settings dialog box to set the scroll range.

[[File] menu (Disassemble panel-dedicated items)]

The following items are exclusive for the [File] menu in the Disassemble panel (other items are common to all the panels).

Note that all these items are invalid during execution of a program.

Save Disassemble Data	Overwrites the contents of the disassembling to the previously saved text file (*.txt)/CSV file (*.csv) (see "(h) Saving the contents of disassembled data "). Note that when the file has never been saved, the same operation is applied as the selection in [Save Disassemble Data As...].
Save Disassemble Data As...	Opens the Data Save dialog box to newly save the contents of the disassembling to the specified text file (*.txt)/CSV file (*.csv) (see "(h) Saving the contents of disassembled data ").
Print...	Opens the Print Address Range Settings dialog box for printing the contents of this panel.

[[Edit] menu (Disassemble panel-dedicated items)]

The following items are exclusive for the [Edit] menu in the Disassemble panel (other items are all invalid).

Copy	When a line is selected, copies the contents of the selected line to the clipboard as a character string. In the case of the edit mode, copies the selected character string to the clipboard.
Rename	Changes to the edit mode to edit the instruction/code at the caret position (see " 2.6.4 Performing line assembly "). This item becomes invalid during execution of a program.
Find...	Opens the Find and Replace dialog box, with its [Find in Files] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Replace in Files] tab selected.
Move...	Opens the Go to the Location dialog box to move the caret to the specified address.

[Context menu]

[Disassemble area and Address area]

Register to Watch1	Registers the selected character string or the word at the caret position to the Watch panel (Watch1) as a watch-expression (the judgment of the word depends on current build tool). At this time, since it is registered as a variable name, the symbol name that is displayed changes depending on the scope.
Register Action Event...	Opens the Action Events dialog box to set an action event to the address at the caret position.
Go to Here	Executes the program from the address indicated by the current PC value to the address corresponding to the line at the caret position. This item becomes invalid during execution of a program/[Build & Download].
Set PC to Here	Sets the address of the line at the current caret position to the current PC value. This item becomes invalid during execution of a program/[Build & Download].
Move...	Opens the Go to the Location dialog box to move the caret to the specified address.
Go to Symbol	Moves the caret to the define position of the selected symbol.
Back to Address	Moves the caret to the position (<i>address</i>) immediately before it is moved by [Go to Symbol]. Note that this item becomes invalid when no symbol name is displayed in the address.

Break Settings	The following cascade menus are displayed to set the break-related event. Note that breakpoints can be set only for lines for which events can be set (see "(1) Event area").
Set Hardware Break	Sets a breakpoint (Hardware Break event) to the address at the caret position (see "2.10.2 Stop the program at the arbitrary position (breakpoint)").
Set Software Break [E1] [E20]	Sets a breakpoint (Software Break event) to the address at the caret position (see "2.10.2 Stop the program at the arbitrary position (breakpoint)").
Set Read Break to [Simulator]	Sets a break event with read access condition to the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "(1) Set a break event (access type) to a variable/I/O register").
Set Write Break to [Simulator]	Sets a break event with write access condition to the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "(1) Set a break event (access type) to a variable/I/O register").
Set R/W Break to [Simulator]	Sets a break event with read/write access condition to the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "(1) Set a break event (access type) to a variable/I/O register").
Set Combination Break [E1] [E20]	Sets a break at the caret position's address or the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break event (see "(1) Set a break event (execution type)").
Set Read Combination Break to [E1] [E20]	Sets a break event by a read access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break (see "(1) Set a break event (access type) to a variable/I/O register").
Set Write Combination Break to [E1] [E20]	Sets a break event by a write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break (see "(1) Set a break event (access type) to a variable/I/O register").
Set R/W Combination Break to [E1] [E20]	Sets a break event by a read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register as the condition for a combination break (see "(1) Set a break event (access type) to a variable/I/O register").
Break Option	Opens the Property panel to set the break function.

Trace Settings	The following cascade menus are displayed to set the trace-related event. Note that events can be set only for lines for which events can be set (see "(1) Event area").
Start Tracing	Sets a Start Tracing event to start collecting the trace data when an instruction of an address at the caret position is executed (see "(1) Setting a trace start event and a trace end event"). [Simulator] In addition, the selecting of the [Use trace function] property in the [Trace] category on the Property panel is automatically set to [Yes].
Stop Tracing	Sets a Stop Tracing event to stop collecting the trace data when an instruction of an address at the caret position is executed (see "(1) Setting a trace start event and a trace end event"). [Simulator] In addition, the selecting of the [Use trace function] property in the [Trace] category on the Property panel is automatically set to [Yes].
Record Reading Value	Sets a Point Trace event to record the access value as the trace data when the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is read accessed (see "(1) When an access to a variable or I/O register occurred").
Record Writing Value	Sets a Point Trace event to record the access value as the trace data when the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is write accessed (see "(1) When an access to a variable or I/O register occurred").
Record R/W Value	Sets a Point Trace event to record the access value as the trace data when the address at the caret or the selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is read/ write accessed (see "(1) When an access to a variable or I/O register occurred").
Record Start R/W Value	Sets a trace event that causes trace recording to start upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(1) Setting a trace start event and a trace end event").
Record End R/W Value	Sets a trace event that causes trace recording to end upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(1) Setting a trace start event and a trace end event").
Show Trace Result	Opens the Trace panel and displays the acquired trace data.
Trace Settings	Opens the Property panel to set the trace function.

Timer Settings [E1] [E20]	The following cascade menus are displayed to set the timer-related event (see "2.14.3 Measuring execution time in a section [E1] [E20]"). Note that events can be set only for lines for which events can be set (see "(1) Event area").
Start timer	Sets a Start Timer event to start measuring the execution time of the program when an instruction of an address at the caret position is executed (see "(a) How to set a timer start event").
Stop timer	Sets a Stop Timer event to stop measuring the execution time of the program when an instruction of an address at the caret position is executed (see "(b) How to set a timer end event").
Set Timer Start R/W Value	Sets a timer start event that causes a measurement of the program's execution time to start upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(a) How to set a timer start event").
Set Timer <N>	Specify a channel ^{Note} in which a timer start event is set.
Set Timer End R/W Value	Sets a timer end event that causes a measurement of the program's execution time to finish upon read/write access to the caret position or a selected variable (global variable, static variable in function, static variable in file) or I/O register (see "(b) How to set a timer end event").
Set Timer <N>	Specify a channel ^{Note} in which a timer start event is set.
View Result of Timer	Opens the Events panel and displays only timer-related events.
Clear Coverage Information [Simulator]	Clears all the coverage measurement results currently being stored in the debug tool. Note that this item appears only when the debug tool used supports the coverage function.
Edit Disassemble	Changes to the edit mode to edit the instruction of the line at the caret position (see "2.6.4 Performing line assembly"). This item becomes invalid during execution of a program.
Edit Code	Changes to the edit mode to edit the code of the line at the caret position (see "2.6.4 Performing line assembly"). This item becomes invalid during execution of a program.
View	The following cascade menus to set the display contents in the disassemble area are displayed.
Show Offset	Displays the offset value of the label. The offset value from the nearest label is displayed when a label is defined for the address.
Show Symbol	Displays the address value in the format "symbol + offset value" (default). Note that when a symbol has been defined as the address value, only the symbol is displayed.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range.
Mixed Display	Sets to the mixed display mode and displays the correspondence between the disassembled data and the source text (default).
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the address at the caret position in this panel.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the memory value corresponding to the address at the caret position in this panel.

Note The specifiable number of channels differs between the RX600 and RX200 Series, as shown below.

RX600 Series: 2 (32 bits * 2) or 1 (64 bits * 1)

RX200 Series: 1 (24 bits * 1)

[Event area] ([E1][E20])

Hardware Break First	The type of break that can be set by a one click operation of the mouse is set as a hardware breakpoint (this is reflected in the setting of the [First using type of breakpoint] property in the [Break] [E1] [E20] category on the Property panel).
Software Break First	The type of break that can be set by a one click operation of the mouse is set as a software breakpoint (this is reflected in the setting of the [First using type of breakpoint] property in the [Break] [E1] [E20] category on the Property panel).

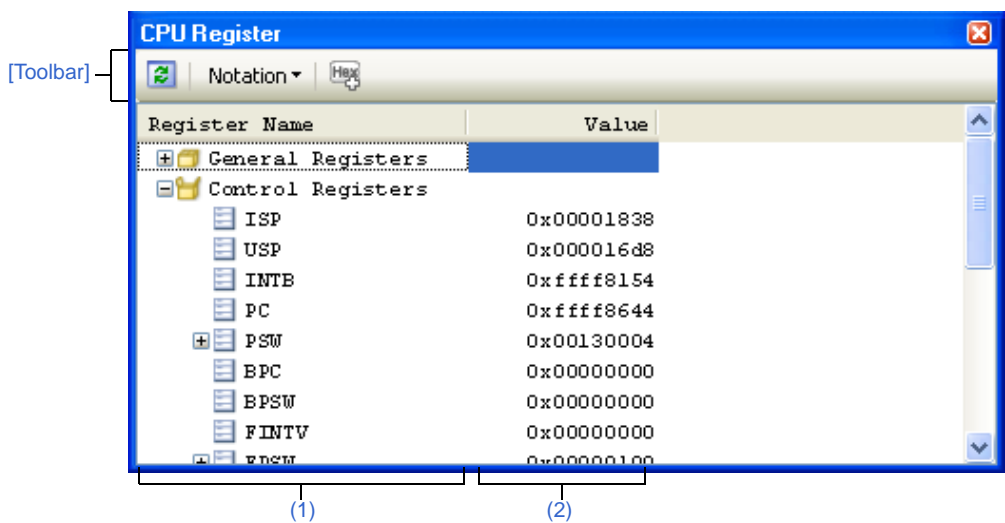
CPU Register panel

This panel is used to display the contents of the CPU register (general-purpose registers and control registers) and change the CPU register values (see "2.11.2 Displaying and changing the CPU registers").

This panel appears only when connected to the debug tool.

Remark When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A-28. CPU Register Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (CPU Register panel-dedicated items)]
- [[Edit] menu (CPU Register panel-dedicated items)]
- [Context menu]

[How to open]



- From the [View] menu, select [CPU Register].



[Description of each area]

(1) [Register Name] area

The types of register are classified as categories (folders), and a list of the respective register names is displayed. Note that neither category names nor register names can be edited and deleted.

The meanings of the icons are as follows.

	Indicates that the register name belonging to this category is displayed. When you double-click on the icon, or click on the "-" mark, the category is closed and the register name is hidden.
	Indicates that the register name belonging to this category is hidden. When you double-click on the icon, or click on the "+" mark, the category is opened and the register name is displayed.

	Indicates the name of the register. When you double-click on the icon, or click on the "+" or "-" marks, the name of the register part is displayed or hidden.
	Indicates the name of the register part.

Category names and register names displayed are as follows (number of "+" marks before register names indicates the depth of the display level).

Table A-2. Register Names in [General Registers] Category [RX]

Register Name (Alias)	Bit Width	Register Name (Alias)	Bit Width
+ R0	32	+ R8	32
+ R1	32	+ R9	32
+ R2	32	+ R10	32
+ R3	32	+ R11	32
+ R4	32	+ R12	32
+ R5	32	+ R13	32
+ R6	32	+ R14	32
+ R7	32	+ R15	32

Table A-3. Register Names in [System Registers] Category [RX]

Register Name	Bit Width	Register Name	Bit Width
+ ISP	32	+ FPSW ^{Note 2}	32
+ USP	32	++ FS	1
+ INTB	32	++ FX	1
+ PC	32	++ FU	1
+ PSW	32	++ FZ	1
++ IPL	4 ^{Note 1}	++ FO	1
++ PM	1	++ FV	1
++ U	1	++ EX	1
++ I	1	++ EU	1
++ O	1	++ EZ	1
++ S	1	++ EO	1
++ Z	1	++ EV	1
++ C	1	++ DN	1
+ BPC	32	++ CE	1
+ BPSW	32	++ CX	1
+ FINTV	32	++ CU	1
		++ CZ	1
		++ CO	1
		++ CV	1
		++ RM	2
		+ ACC	64

Notes 1. The bit width is 3 bits for the RX610 Group.

2. FPSW register is not supported by the RX210 Group.

This area is provided with the following functions.

(a) Registering watch expression

CPU registers/categories can be registered in the [Watch panel](#) as watch expressions.

See "(1) [Registering watch expressions](#)" for details on how to operate it.

Remarks 1. When you have registered a watch expression with a category as the object, all of the CPU registers belonging to that category are registered as watch expressions.

2. A scope specification is automatically added to a registered watch expression.

(2) [Value] area

The values of each CPU register are displayed and changed.

The radix of a data value can be selected by the button on the toolbar or the context menu item. In addition, a display format adding the value in hexadecimal number constantly can also be selected as well.

The meanings of the colors of the CPU register values are as follows (character colors and background colors depend on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#)).

Display Example (Default)			Description
0x0	Character color	Blue	The value of the CPU register that the user is changing Press the [Enter] key to write to the target memory.
	Background color	Standard color	
0x0	Character color	Brown	The value of the CPU register that has been changed because of the execution of a program The highlighting is rest by executing again the program.
	Background color	Cream	

This area is provided with the following functions.

(a) Changing the CPU register value

To edit the CPU register value, change the value directly from the keyboard after double-clicking on the value to be edited (press the [Enter] key to cancel the edit mode).












After you edit the value of the CPU register, it is written to the target memory of the debug tool by pressing the [Enter] key or moving the focus to outside the edit region.

(b) Saving the contents of the CPU register

The [Save As dialog box](#) can be opened by selecting the [File] menu >> [Save CPU Register Data As...], and all the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).

See "(4) [Saving the displayed CPU register contents](#)" for details on the method for saving the contents of the CPU register.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel. This item becomes invalid during execution of a program.
Notation	The following buttons to change the notation of a data value are displayed.
	Displays the value of the selected item (including sub-items) in the default notation (default).
	Displays the value of the selected item (including sub-items) in hexadecimal number.
	Displays the value of the selected item (including sub-items) in signed decimal number.
	Displays the value of the selected item (including sub-items) in unsigned decimal number.
	Displays the value of the selected item (including sub-items) in octal number.
	Displays the value of the selected item (including sub-items) in binary number.
	Displays the character string of the selected item (including sub-items) in ASCII code. If the character size is 2 bytes and above, it is displayed with the characters for each 1 byte arranged side-by-side.
	Displays the value of the selected item in Float. Note that when the value is not 4-byte data, displays it in the default notation.
	Displays the value of the selected item in Double. Note that when the value is not 8-byte data, displays it in the default notation.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value.

[[File] menu (CPU Register panel-dedicated items)]

The following items are exclusive for the [File] menu in the CPU Register panel (other items are common to all the panels).

Note that all these items are invalid during execution of a program.

Save CPU Register Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the CPU register"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save CPU Register Data As...].
Save CPU Register Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the CPU register").

[[Edit] menu (CPU Register panel-dedicated items)]

The following items are exclusive for [Edit] menu in the CPU Register panel (other items are all invalid).

Cut	Deletes the selected character string and copies it to the clipboard. This item becomes valid only when the character string is being edited.
Copy	Copies the selected character string to the clipboard during editing. If a line is selected, copies the register or the category to the clipboard. The copied item can be pasted to the Watch panel .
Paste	Pasts the character string copied in the clipboard to the caret position. This item becomes valid only when the character string is being edited.
Select All	Selects all the items of this panel.
Find...	Opens the Find and Replace dialog box, with its [Find in Files] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Replace in Files] tab selected.

[Context menu]

Register to Watch1	Registers the selected register or category to the Watch panel (Watch1).
Copy	Copies the selected character string to the clipboard during editing. If a line is selected, copies the register or the category to the clipboard. The copied item can be pasted to the Watch panel .
Notation	The following cascade menus to specify the notation of a data value are displayed.
AutoSelect	Displays the value of the selected item (including sub-items) in the default notation (default).
Hexadecimal	Displays the value of the selected item (including sub-items) in hexadecimal number.
Signed Decimal	Displays the value of the selected item (including sub-items) in signed decimal number.
Unsigned Decimal	Displays the value of the selected item (including sub-items) in unsigned decimal number.
Octal	Displays the value of the selected item (including sub-items) in octal number.
Binary	Displays the value of the selected item (including sub-items) in binary number.
ASCII	Displays the character string of the selected item (including sub-items) in ASCII code. If the character size is 2 bytes and above, it is displayed with the characters for each 1 byte arranged side-by-side.
Float	Displays the value of the selected item in Float. Note that when the value is not 4-byte data, displays it in the default notation.
Double	Displays the value of the selected item in Double. Note that when the value is not 8-byte data, displays it in the default notation.
Include Hexadecimal Value	Adds the value in hexadecimal number enclosing with "()" at the end of the value.

IOR panel

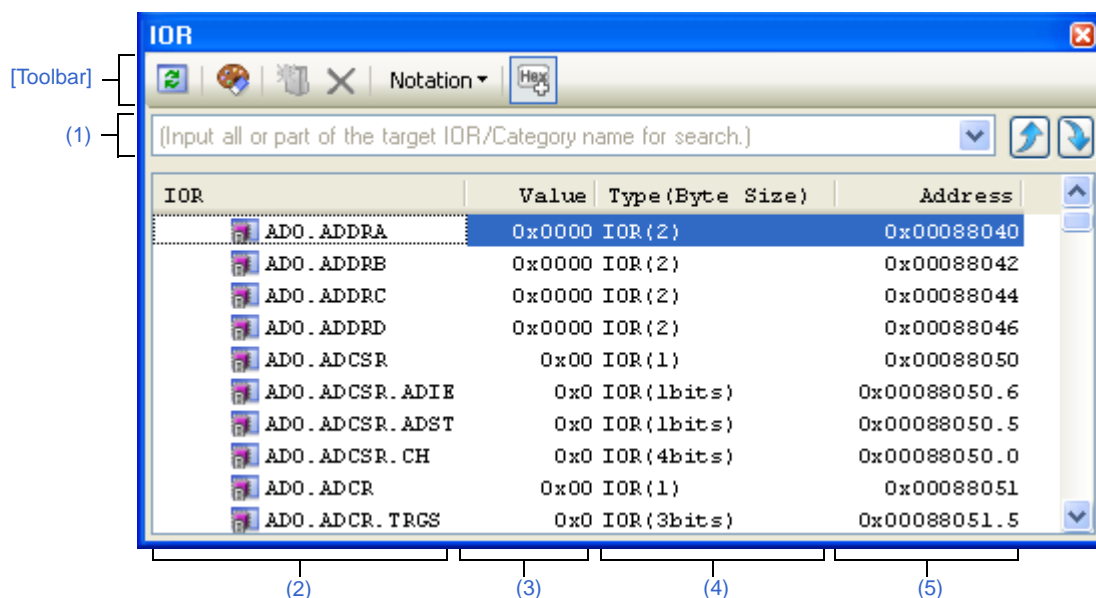
This panel displays the contents of I/O registers and changes their values (see Section "2.11.3 Displaying and changing the I/O registers").

Note that this panel can only be opened when CubeSuite+ is connected with the debug tool.

Caution The I/O registers that get the microcontroller actuated by a read operation are protected against reads, so that no values are read from those registers ([Value] marked with "?").
To obtain the contents of the I/O registers protected against reads, select [Forcibly Load Values] from the context menu.

Remark By double-clicking a line delimiting each area on panel, it is possible to change the relevant area to the smallest displayable width without omitting the content in it.

Figure A-29. IOR Panel



This section describes the following.




- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (IOR Panel-Only Items)]
- [[Edit] menu (IOR Panel-Only Items)]
- [Context Menu]



[How to open]

- Choose [IOR] from the [View] menu.

[Description of each area]**(1) Search area**

This area performs a search of I/O register names.




	Specify the character string to be searched (not case-sensitive). Enter a character string directly from the keyboard (specifiable in up to 512 characters) or select an input history item from the drop-down list (up to 10 history entries).
	Searches for I/O register names that include the character string specified in the text box in upward direction, with the search result placed in selected state.
	Searches for I/O register names that include the character string specified in the text box in downward direction, with the search result placed in selected state.

- Remarks 1.** The I/O register names that are hidden as classified by category (folder) also are searched (expanded and placed in selected state).
- 2.** After entering the character string to be searched, press the [Enter] key and the same option as would be when the  button is clicked is performed, or press the [Shift] + [Enter] keys and the same operation as would be when the  button is clicked is performed.

(2) [IOR] area

This area displays I/O register names in list form as classified by type of I/O register as category (folder).

The meaning of each displayed icon is as follows.


	Indicates that names of I/O registers in this category are currently displayed. Double-clicking this icon or clicking "-" will close the category and hide the names of corresponding I/O registers. Note that categories are, by default, nonexistent. If necessary, create a new category and then Edit the tree .
	Indicates that names of I/O registers in this category are currently hidden. Double-clicking this icon or clicking "+" will open the category and display the names of corresponding I/O registers. Note that categories are, by default, nonexistent. If necessary, create a new category and then Edit the tree .
	Displays I/O register names.


Remark By clicking the header part of this area, it is possible to sort category names in order of character code.
(The I/O register names in each category are sorted in the same way.)

This area has the following features.

(a) Edit the tree

The tree form can be edited by classifying each I/O register by any category (folder).

To create a new category, move the caret to the I/O register name for which a category is to be created and click the  button in the toolbar or select [Create Category] from the context menu. Then enter any category name (specifiable in up to 1,024 characters).

To remove a category, select the category to be removed and click the  button in the toolbar or select [Remove] from the context menu. Note, however, that only blank categories can be removed.

Also, to edit a category name, select the category name to edit and follow one of the following procedures:

- Click the category name again and edit it directly from the keyboard

- Choose [Change Name] from the [Edit] menu and then edit the category name directly from the keyboard
- Press the [F2] key and then edit the category name directly from the keyboard

When a category is created, drag-and-drop I/O register names directly into the category. That way, I/O register names can be displayed in tree form classified by category.

Similarly, the order in which categories or I/O register names are displayed (one above or below another) can be freely changed by a drag-and-drop operation.

- Cautions 1. A category cannot be created within another category.**
2. I/O registers cannot be added or removed.

(b) Registration of watch expressions

I/O registers or categories can be registered as watch expressions on the [Watch panel](#).

For details on how to do it, see "(1) [Registering watch expressions](#)".


- Remarks 1.** If a watch expression is registered for a category, all of the I/O registers belonging to that category are registered as watch expressions.
2. The registered watch expressions have their scope specification automatically given.

(3) [Value] area

This area displays or changes I/O register values.

The notation (numeral representation) in which values are displayed can be selected using the appropriate toolbar button or selecting from the context menu. Also, it is possible to select a display form that always adds hexadecimal equivalents to the ordinary display.

The meaning of marks displayed as I/O register values and their colors are as follows (The colors in which text and backgrounds are displayed depend on how the [\[General - Font and Color\] category](#) of the [Option dialog box](#) is set.).

Example display (default)			Description
0x0	Text color	Blue	The value of the I/O register that the user is changing (press the [Enter] key to write to the target memory).
	Background color	Standard color	
0x0	Text color	Brown	The value of the I/O register that has been changed because of the execution of a program To reset the highlighting, select the  button on the toolbar or [Reset Color] from the context menu.
	Background color	Cream	
?	Text color	Gray	Values of I/O registers protected against read ^{Note}
	Background color	Standard color	

Note This refers to the I/O registers that get the microcontroller actuated by a read operation.

To read the value of read-protected I/O register, select [Force Read Value] from the context menu.

Caution The 1-byte or 2-byte I/O registers and the 1-bit I/O registers mapped to those 1-byte or 2-byte I/O registers differ in timing with which values are retrieved. Therefore, while a value from the same I/O register is being displayed, it is possible that the displayed value is different.

Remark The values are sorted in ascending order of numeric value by clicking the header part of this area.

This area has the following features.

(a) Alteration of I/O register values

When changing I/O register values, select the target I/O register value and then click it again to edit it directly from the keyboard. (Pressing the [Esc] key cancels the edit mode.)

After editing an I/O register value, hit the [Enter] key or move the focus to other than the edit area. The edited value is written into the debug tool's target memory.

For details on how to edit I/O register values, see "(4) Changing the contents of I/O registers".

(b) Saving of I/O register values

Choosing [Save IOR Data As ...] from the [File] menu opens the [Save As dialog box](#), making it possible to save all content of this panel to a text file (*.txt) or CSV file (*.csv).

For details on how to save I/O register values, see "(6) Saving the displayed I/O register contents".

(4) [Type (Byte Size)] area

This area displays the type information of each I/O register in the form shown below.

- <Type of I/O register> [<Access attribute> <All accessible sizes>](<Size>)

Access attribute	One of the following is displayed as access attribute.	
	R	Read only
	W	Write only
	R/W	Read/Write
All accessible sizes	Accessible sizes, in bit units, are enumerated in increasing order by separating each with a comma ",".	
Size	Shows the size of each I/O register. If displayable in byte units, the size is shown in bytes, if displayable in bit units, the size is shown in bits, with the respective units given.	

Examples 1. For the "IOR [R/W 1.8] (1 byte)" case

This refers to an I/O register that is read/writable, accessible in 1 bit and 8 bits, and 1 byte in size.

2. For the "IOR [R/W 1] (1 bit)" case

This refers to an I/O register that is read/writable, accessible in 1 bit, and 1 bit in size.

Remark The type information is sorted in order of character code by clicking the header part of this area.

(5) [Address] area

This area displays the addresses to which the I/O registers are mapped (always shown in hexadecimal).

However, the bit registers displayed here are given bit offset values, as shown in the examples below.

Examples 1. For the "0xFF40" case












This register is mapped to "0xFF40".

2. For the "0xFF40.4" case

This is a bit register mapped to bit 4 of the address "0xFF40".

Remark The addresses are sorted in ascending order of numeric value by clicking the header part of this area.

[Toolbar]

	Obtains latest information from the debug tool and updates the display. No data are reloaded for the I/O registers protected against read. However, this button is disabled during program execution.
	Resets the highlighting for a selected I/O register, which indicates that its value has changed as a result of program execution. However, this button is disabled during program execution.
	Adds a new category (folder). Enter a category name directly in the text box. While you can create any number of new categories, note that a category cannot be created within another category. However, this button is disabled during program execution.
	Deletes a selected range of character string. If a blank category is in selected state, the category is deleted (I/O registers cannot be deleted.).
Notation	Shows the following buttons that change the form in which values are displayed.
	Displays the value of a selected item in hexadecimal (default).
	Displays the value of a selected item in signed decimal.
	Displays the value of a selected item in unsigned decimal.
	Displays the value of a selected item in octal.
	Displays the value of a selected item in binary.
	Displays the value of a selected item in ASCII code.
	Adds a hexadecimal equivalent for the displayed value of a selected item at the end of it, with the equivalent enclosed in parentheses ().

[[File] menu (IOR Panel-Only Items)]

The [File] menu used exclusively for the IOR panel is as follows (The other items are shared.).

However, all of these items are disabled during program execution.

Save IOR Data	Saves the content of this panel to a text file (*.txt) or CSV file (*.csv) that has been saved previously (see "(b) Saving of I/O register values"). If this item is selected for the first time after startup, the same operation as [Save IOR Data As ...] would have been selected is performed.
Save IOR Data As...	Opens the Save As dialog box in order to save the content of this panel to a specified text file (*.txt) or CSV file (*.csv) (see "(b) Saving of I/O register values").

[[Edit] menu (IOR Panel-Only Items)]

The [Edit] menu used exclusively for the IOR panel is as follows (All other items are disabled.).

Cut	Cuts a selected range of character string and moves it to the clipboard (No I/O registers and categories can be cut.).
Copy	Copies a selected range of character string to the clipboard. If an I/O register or category is in selected state, that item is copied. Note that the copied item can be pasted to the Watch panel .
Paste	Pastes the content of the clipboard to the caret position when text is in editing mode (I/O registers and categories cannot be pasted.).

Delete	Deletes a selected range of character string. If a blank category is in selected state, that item is deleted (No I/O registers can be deleted.).
Select All	Selects all character strings whose text is in edit mode. If the text is in other than edit mode, all I/O registers and categories are placed in selected state.
Rename	Edits the name of a selected category.
Find...	Moves the focus to the text box in the search area.
Go To...	Opens the Go to the Location dialog box to move the caret to a specified I/O register.

[Context Menu]

Register to Watch1	Registers the selected I/O register or category to the Watch panel (Watch1).
Refresh	Obtains latest information from the debug tool and updates the display. No data are reloaded for the I/O registers protected against read. However, this button is disabled during program execution.
Force Read Value	Forcibly loads a value once for I/O registers protected against read.
Move...	Opens the Go to the Location dialog box .
Create Category	Adds a new category (folder). Enter a category name directly into the text box. While you can create any number of new categories, note that a category cannot be created within another category. However, this button is disabled during program execution.
Copy	Copies a selected range of character string to the clipboard. If an I/O register or category is in selected state, that item is copied. Note that the copied item can be pasted to the Watch panel .
Delete	Deletes a selected range of character string. If a blank category is in selected state, that item is deleted (No I/O registers can be deleted.).
Notation	Shows the following cascaded menu to specify the form in which values are displayed.
Hexadecimal	Displays the value of the selected item in hexadecimal (default).
Signed Decimal	Displays the value of the selected item in signed decimal.
Unsigned decimal	Displays the value of the selected item in unsigned decimal.
Octal	Displays the value of the selected item in octal.
Binary	Displays the value of the selected item in binary.
ASCII	Displays the value of the selected item in ASCII code.
Include Hexadecimal Value	Adds a hexadecimal equivalent for the displayed value of a selected item at the end of it, with the equivalent enclosed in parentheses ().
Reset Color	Resets the highlighting for a selected I/O register, which indicates that its value has changed as a result of program execution.

Local Variables panel

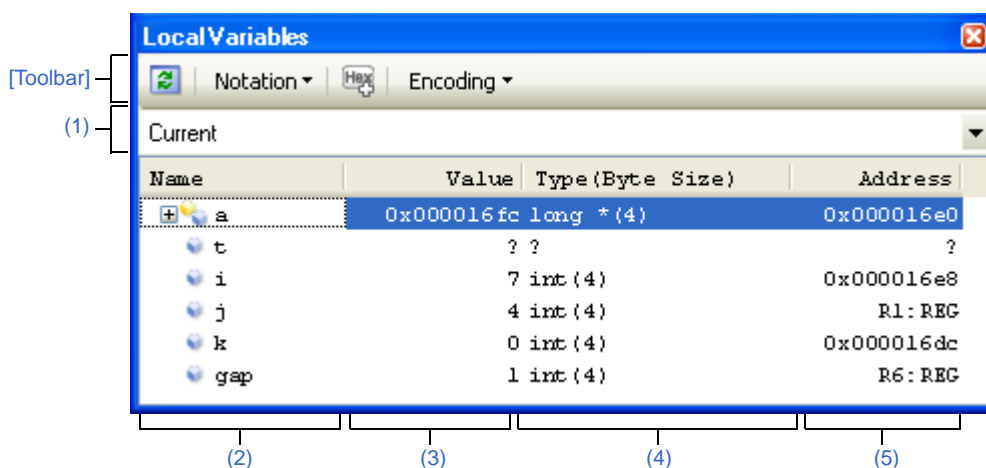
This panel displays the contents of local variables and changes their values (see "2.11.5 Displaying and changing local variables").

Note that this panel can only be opened when CubeSuite+ is connected with the debug tool.

- Cautions 1.** During program execution, nothing is displayed on this panel. Each area on this panel are displayed at the time the program has stopped running.
- 2.** Because of optimization by a compiler, for blocks where variables, or the subject to be operated on, are not in use, there may be no variable data in the stack and registers. In this case, no variables are displayed that are the subject to be operated on.

Remark By double-clicking a line delimiting each area on panel, it is possible to change the relevant area to the smallest displayable width without omitting the content in it.

Figure A-30. Local Variables panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Local Variable Panel-Only Items)]
- [[Edit] menu (Local Variable Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Local Variables] from the [View] menu.

[Description of each area]

(1) Scope area

This area selects the scope of local variables to be displayed from the drop-down list.




The selectable items are as follows.

Item	Operation
Current	Displays local variables at scope of the current PC value.
<depth> <Function name() [file name#line number]> ^{Note}	Displays local variables at scope of the calling function. After the program execution, the scope selected here is retained as long as the selected scope exists.

Note It shows the function caller displayed on [Call Stack panel](#).

(2) [Name] area

Displays local variable names or function names. Function parameters are also displayed as local variables. Also, arrays, pointer-type variables, and structures/unions have their hierarchical structure displayed in tree form. This area cannot be edited. The meaning of each icon displayed here is as follows.

	Denotes a variable. Auto variables, internal static variables, and register variables are also displayed ^{Note} . Arrays, pointer-type variables, and structures/unions have their hierarchical structure displayed in tree form. If a variable is marked with "+" at the beginning, clicking it expands the next part of the variable (After expansion, the mark changes to "-").	
	Array	All elements in the array
	Pointer-type variables	Variable at the destination pointed to by the pointer Note that if the destination pointed to by the pointer is a pointer, it is further assigned a "+" mark. Clicking it shows the other part that is referenced. However, if the value pointed to by the pointer is unknown, it is marked with "?".
	Structure/union	All members of the structure/union
	Denotes a parameter.	
	Denotes a function.	

Note To display auto variables, be aware that the exact values of local variables cannot be displayed at the prolog of functions ("{" in functions) and epilog of functions (")" in functions). (The addresses of auto variables are relative addresses from the stack pointer (SP), so that they are indefinite until the SP value is fixed in the function. As the SP is manipulated at the prolog and epilog, its value is indefinite. Therefore, exact values cannot be displayed at the prolog and epilog.)

This area has the following features.

(a) Registration of watch expressions

C variables can be registered in the [Watch panel](#) as watch expressions.
For details on how to register, See "(1) [Registering watch expressions](#)".

Remark The registered watch expressions have their scope specification automatically given.

(b) Jump to memory

Selecting [Jump to Memory] on context menu opens the [Memory panel](#) (Memory1) with the caret moved to the address at which a selected local variable is located (If the memory panel (Memory1) is already open, CubeSuite+ jumps to it directly.).

(3) [Value] area

This area displays or changes the values of local variables.

The notation of a data value and encoding of character strings can be selected using the toolbar buttons or from the context menu. Also, it is possible to select a display form where hexadecimal display is always added.

The meaning of marks displayed as values of local variables and their colors are as follows (The colors in which text and backgrounds are displayed depend on how the [\[General - Font and Color\] category](#) of the [Option dialog box](#) is set).

Example display (default)			Description
0x0	Text color	Blue	Local variable values that have been changed by the user. Press [Enter] to write them to the target memory.
	Background color	Standard color	
0x0	Text color	Brown	Local variable values that have been changed due to program execution Note Executing the program again will reset the highlighting color.
	Background color	Cream	
?	Text color	Gray	Local variable values that could not be acquired.
	Background color	Standard color	

Note Applicable only to cases where the displayed variable names remained same from the program start point to the break point while their values were changed.

This area has the following features.

(a) Alteration of local variable and parameter values

To change a local variable value and parameter value, select the local variable value to be changed first and, after clicking it again, enter directly new values from the keyboard. (Pressing the [Esc] key cancels the edit mode.)

The edited local variable values and parameter values are written into the debug tool's target memory by pressing the [Enter] key or moving the focus to other than the edit area.

For details on how to change local variable and parameter values, see ["\(2\) Changing the contents of local variables"](#).

(b) Saving of local variable values

Choosing [Save Local Variables Data As ...] from the [File] menu opens the [Save As dialog box](#), making it possible to save all content of this panel to a text file (*.txt) or CSV file (*.csv).

For details on how to save local variable values, see ["\(3\) Saving the displayed contents of local variables"](#).

(4) [Type (Byte Size)] area

This area displays the type names of local variables. They are displayed in forms conforming to C language description.

Arrays have the number of their elements added in "[]" and functions have their size (in bytes) added in "()" when they are displayed.

Note that this area cannot be edited.

















(5) [Address] area

This area displays the addresses of local variables. If variables are assigned to registers, the register names are displayed.

This area cannot be edited.

[Toolbar]

Each button in the toolbar are disabled during program execution.

	Obtains latest information from the debug tool and updates display.
Notation	Shows the following buttons that change the form in which values are displayed.
	Displays values on this panel in per-variable predetermined notation (default).
	Displays values on this panel in hexadecimal.
	Displays values on this panel in decimal.
	Displays values on this panel in octal.
	Displays values on this panel in binary.
	Displays array indexes on this panel in decimal (default).
	Displays array indexes on this panel in hexadecimal.
	Displays values on this panel in Float. Note that non-4 byte data and data with type information are displayed in default notation.
	Displays values on this panel in Double. Note that non-8 byte data and data with type information are displayed in default notation.
	Adds the hexadecimal equivalent in bracket "()" at the end of the value.
Encoding	Shows the following buttons that change the encode in which string variables are displayed.
	Displays string variables in ASCII code (default).
	Displays string variables in Shift_JIS code.
	Displays string variables in EUC-JP code.
	Displays string variables in UTF-8 code.
	Displays string variables in UTF-16 code.

[[File] menu (Local Variable Panel-Only Items)]

The [File] menu items listed below are provided exclusively on the Local Variables panel (Other menu items are shared with other panels.).

However, all of these items are disabled during program execution.

Save Local Variables Data	Overwrites the last-saved text file (*.txt) or CSV file (*.csv) with the contents of this panel (see "(b) Saving of local variable values "). If you select this item for the first time after the start of the program, the operation will be the same as selecting [Save Local Variables Data As...].
Save Local Variables Data As...	Opens the Save As dialog box in order to save the contents of this panel to a specified text file (*.txt) or CSV file (*.csv) (see "(b) Saving of local variable values ").

[[Edit] menu (Local Variable Panel-Only Items)]

The [Edit] menu items listed below are provided exclusively on the Local Variables panel. (All the other menu items are disabled.)

Copy	Copies the content of a selected line or a character string to the clipboard.
Select All	Puts the item into all selected state.
Rename	Goes to edit mode in order to change the value of a selected local variable (see "(2) Changing the contents of local variables "). However, this item is disabled during program execution.
Find...	Opens the Find and Replace dialog box, with the [Find in Files] tab selected.
Replace...	Opens the Find and Replace dialog box, with the [Replace in Files] tab selected.

[Context menu]

All the items in the context menu are disabled during program execution.

Register to Watch1	Registers a selected local variable in the Watch panel (Watch1).
Copy	Copies the content of a selected line or a character string to the clipboard.
Notation	Shows the following cascaded menu to specify the form in which values are displayed.
AutoSelect	Displays values on this panel in per-variable predetermined notation (default).
Hexadecimal	Displays values on this panel in hexadecimal.
Decimal	Displays values on this panel in decimal.
Octal	Displays values on this panel in octal.
Binary	Displays values on this panel in binary.
Decimal Notation for Array Index	Displays array indexes on this panel is decimal (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel is hexadecimal.
Float	Displays values on this panel in Float. Note that non-4 byte data and data with type information are displayed in default notation.
Double	Displays values on this panel in Double. Note that non-8 byte data and data with type information are displayed in default notation.
Include Hexadecimal Value	Adds the hexadecimal equivalent in bracket "()" at the end of the value.
Encoding	Shows the following cascaded menu to specify character code.
ASCII	Displays string variables in ASCII code.
Shift_JIS	Displays string variables in Shift_JIS code (default).
EUC-JP	Displays string variables in EUC-JP code.
UTF-8	Displays string variables in UTF-8 code.
UTF-16	Displays string variables in UTF-16 code.
Jump to Memory	Opens the Memory panel (Memory1), with the caret moved to the address indicated by the selected line.

Watch panel

This panel is used to display the contents of the registered watch expressions and change their values (see "2.11.6 Displaying and changing watch expressions").

Up to a maximum of four of these panels can be opened. Each panel is identified by the names "Watch1", "Watch2", "Watch3", and "Watch4" on the titlebar, and the watch expressions can be registered/deleted/moved individually.

Watch expressions can be registered in this panel as well as in the [Editor panel](#), [Disassemble panel](#), [Memory panel](#), [CPU Register panel](#), [Local Variables panel](#) or [IOR panel](#).

When the panel is closed with registered watch expressions, the panel closes but the information on the registered watch expressions is retained. Therefore, if the same panel is opened again, it is opened with the watch expressions registered.

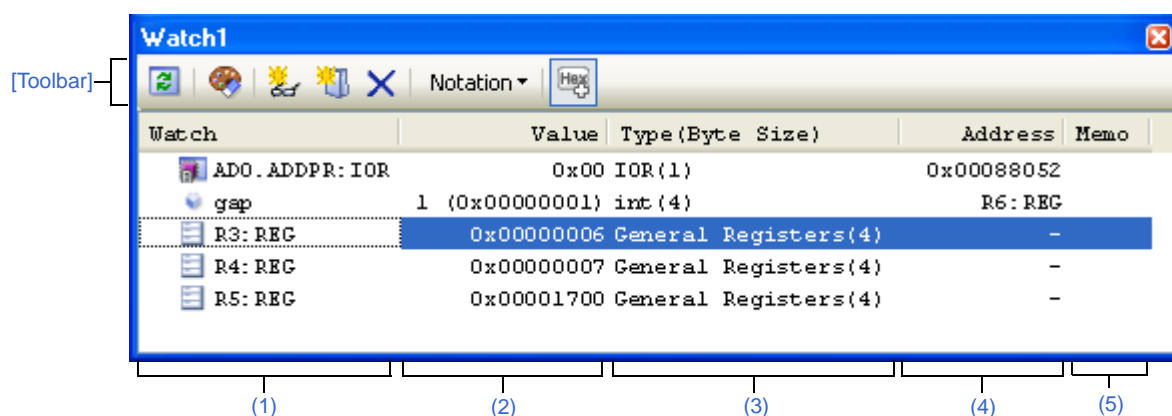
The display contents are automatically updated when the value of the watch expression changes after a program is executed (when the execution is done in steps, the display is updated after each step).

In addition, by enabling the [Real-time display update function](#), it is also possible to update the display contents in real-time even while a program is being executed.

This panel appears only when connected to the debug tool.

Remark When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A-31. Watch Panel



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[\[File\] menu \(Watch panel-dedicated items\)\]](#)
- [\[\[Edit\] menu \(Watch panel-dedicated items\)\]](#)
- [\[Context menu\]](#)

[How to open]

- From the [\[View\]](#) menu, select [\[Watch\]](#) >> [\[Watch 1 - 4\]](#).









[Description of each area]**(1) [Watch] area**

All the registered watch expressions are displayed in a list.

Clicking the title of the list in this area sorts the watch expressions in the list in alphabetical order.

Categories (folders) can be created to categorize the watch expressions and display them in the tree view (see "(a) Tree editing").


The meanings of the icons are as follows.


	Indicates that the watch expression belonging to this category is displayed. When you double-click on the icon, or click on the "-" mark, the category is closed and the watch expression is hidden.
	Indicates that the watch expression belonging to this category is hidden. When you double-click on the icon, or click on the "+" mark, the category is opened and the watch expression is displayed.
	Indicates that the watch expression is a variable. The "+" "/" "-" mark is shown at the top of each watch expression that represents an array, pointer type variable, structure/union/class, or register. Click the mark to Expand/shrink display .
	Indicates that the watch expression is a function.
	Indicates that the watch expression is an immediate value.
	Indicates that the watch expression is an expression.
	Indicates that the watch expression is an I/O register.
	Indicates that the watch-expression is a CPU register. The "+" "/" "-" mark is shown at the top of each watch expression that has the lower level register (part of the register). Click the mark to Expand/shrink display .

This area is provided with the following functions.

(a) Tree editing

Watch expressions can be categorized (by folders) and displayed in the tree view.

To create a category, click the  button on the toolbar or select [Create Category] from the context menu after moving the caret to the position to create a category, and then input a desired name from the keyboard.

To delete a category, select the category then click the  button on the toolbar or select [Delete] from the context menu.

To rename the created category, select the category then do either one of the following.

- Click the name again, then directly rename the category name.
- Select the [Edit] menu >> [Rename], then directly rename the category name.
- Press the [F2] key, then directly rename the category name.

By directly dragging and dropping the registered watch expression in the created category, each category is displayed in the categorized tree view.

Also, the display order of the categories and the watch expressions (upper or lower position) can be changed easily by drag and drop operation.

Cautions 1. Categories cannot be created within categories.

2. Up to 64 categories can be created in one watch panel (if this restriction is violated, a message appears).

Remark Drag and drop the watch expressions/categories in other watch panel (Watch1 to Watch4) to copy them.

(b) Expand/shrink display

At the top of the watch expression represents arrays, pointer type variables, structures/unions/classes, and registers (with the name of the part), "+"/"-" mark is displayed. Click the mark to expand the contents ("+" mark is changed to "-" after the expansion).

Watch Expression	Contents When Expanded
Array	All elements in the array Select [Encoding] >> [ASCII] from the context menu to display the value as a string (up to 256 characters). Note, however, that any characters that cannot be displayed in the encoding will be shown as periods "." or "?".
Pointer type variable	Variables that are pointed to
Structure/Union/class	All member of the structure/union/class
Register	Name of the bit/bit string that constructs the register Example) PSW register IPL register, PM register, U register, I register O register, S register, Z register, C register

(c) Registering new watch expression

There are three ways as follows to register new watch expressions.

<1> Register from other panels

Do either one of the following to register watch expressions in other panels.

- Drag and drop the target character string in this area in the desired watch panel (Watch1 to Watch4). Note that this operation is not valid in the [Editor panel](#).
- Select [Register to Watch1] from the context menu after selecting the target character string or place the caret on either of the target character string (the target is automatically determined).
- Select the [Edit] menu >> [Paste] in this area in the desired watch panel (Watch1 to Watch4) after selecting the [Edit] menu >> [Copy] for the target character string.

The relationship between panels that can use this operation and targets that can be registered as watch expressions is as follows.


Table A-4. Relationship between Panels and Targets That Can be Registered as Watch Expressions

Panel Name	Targets That can be Registered as Watch Expressions
Editor panel	C/C++ language variables ^{Note 1} , CPU registers, I/O registers, and assembler symbols
Disassemble panel	C/C++ language variables ^{Note 1} , CPU registers, I/O registers, and assembler symbols
CPU Register panel	CPU registers ^{Note 2}
Local Variables panel	C language variables ^{Note 1} (local variables)
IOR panel	I/O registers ^{Note 2}

Notes 1. C89, C99, or C++ language variables.

2. The scope-specification is automatically added to the registered watch expression.

<2> Directly register in the Watch panel

Click the  button on the toolbar or select [Add New Watch] from the context menu in the desired watch panel (Watch1 to Watch4) to display an entry box for a new watch expression in the bottom of this area.

Directly input a watch expression from the keyboard in the [Watch] area in the entry box then press the [Enter] key.

The input format of the watch expression is as follows:

Table A-5. Input Format of Watch Expression

Watch Expression	Value to be Displayed
Name of C/C++ language variable ^{Note 1}	Value of a C/C++ language variable
<i>Watch expression</i> [<i>Watch expression</i>]	Element of an array
<i>Watch expression</i> . Member name ^{Note 2}	Member of a structure/union/class member
<i>Watch expression</i> -> Member name ^{Note 2}	Member of a structure/union/class member that is pointed to
<i>Watch expression</i> . * <i>Cast expression</i>	Value of the pointer to a member variable
<i>Watch expression</i> -> * <i>Cast expression</i>	Value of the pointer to a member variable
* <i>Watch Expression</i>	Value of a pointer variable
& <i>Watch Expression</i>	Location address
(Type name) <i>Watch expression</i>	Value cast into a specified type
Name of a CPU register	Value of the CPU register
I/O register name	I/O register value
Label name ^{Note 3} , EQU symbol name ^{Note 3} and [immediate value]	Values of a label, an EQU symbol, and immediate address
Integer constant	Integer constant value
Floating constant	Floating constant value
Character constant	Character constant value

Notes 1. C89, C99, or C++ language variables.

2. When using a member variable of a base class, specify the scope before the member name. (e.g. variable.BaseClass::member).

3. If the label name or EQU symbol name includes a "\$," be sure to enclose the name in "{ }". (Example: {\$Label!})

Any imaginary number must be multiplied by an uppercase "I" (e.g. 1.0 + 2.0*I). When you specify the CPU register name "I", add ":REG" (e.g. I:REG) to distinguish it from the keyword "I".

Watch expressions can be registered with specifying the scope. The scope specifications with watch expression registration are as follows:

Table A-6. Scope Specification of C/C++ Language Function Used with Watch Expression Registration

Scope Specification	Load Module File Name ^{Note 1}	Source File Name ^{Note 1}	Function Name ^{Note 2}	Search target
prog\$file#func	prog	file	func	Static function
prog\$func	prog	global	func	Global function
file#func	current	file	func	Static
func	current	current	func	All ^{Note 2}

- Notes 1.** If the load module name or file name includes a space or one of the following symbols, be sure to enclose the name in double-quotes (" "). (Example: "c:\folder\prog.abs"\$file.c#func)
 \$, #, (,), [,], &, ^, ~, %, +, -, *, /, :, ;, ', |, \, <, >, !
- 2.** Static and global functions are searched for in that order from the scope of the current PC value. Static functions outside the scope are not searched.
 When you specify a function defined within a name space, add the scope to the function (e.g. Scope::func). If there are two or more functions that have the same name, also specify the parameter type (e.g. func(int, int)).

Table A-7. Scope Specification of C/C++ Language Variable Used with Watch Expression Registration

Scope Specification	Load Module File Name ^{Note 1}	Source File Name ^{Note 1}	Function Name ^{Note 2}	Function Name ^{Note 2}	Search target
prog\$file#func#var	prog	file	func	var	Static variables in a static function ^{Note 2,3}
prog\$file#var	prog	file	global	var	Static variables in a file
prog\$var	prog	global	global	var	Global variable
file#func#var	current	file	func	var	Static variables in a static function ^{Note 2,3}
file#var	current	file	global	var	Static variables in a file
var	current	current	current	var	All ^{Note 4}

- Notes 1.** If the load module name or file name includes a space or one of the following symbols, be sure to enclose the name in double-quotes (" ") (Example: "c:\folder\prog.abs"\$file.c#func).
 \$, #, (,), [,], &, ^, ~, %, +, -, *, /, :, ;, ', |, \, <, >, !
- 2.** When you specify a function defined within a name space, add the scope to the function (e.g. Scope::func). If there are two or more functions that have the same name, also specify the parameter type (e.g. func(int, int)).
- 3.** If the current PC value exists within a specified function, even the local variables that are not declared to be static are searched for.
- 4.** Local variables, intra-file static variables and global variables are searched for in that order from the scope of the current PC value. Local variables and intra-file static variables outside the scope are not searched.

Table A-8. Scope Specification of CPU Register with Watch Expression Registration

Scope Specification	Register Bank	Name of CPU Register
R10:REG	-	R10

Table A-9. Scope Specification of I/O register with Watch-Expression Registration

Scope Specification	Name of I/O register
AD0.ADCR:IOR	AD0.ADCR
AD0.ADCR	AD0.ADCR

- Remarks 1.** By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 Symbol name completion function").
2. An immediate value is treated as a value. Note, however, that an immediate value with operators can be used.
 3. An arithmetic expression with symbols can be used for a watch expression.
 4. If the same name exists either in C/C++ language variables, CPU registers or I/O registers, and it is registered without specifying scopes, then its value will be displayed after the symbol is determined in the following order.
Variable of C language > CPU registers > I/O register
 5. If a local variable and a global variable exist with the same name, and its symbol name is registered without specifying scopes, then its value will be displayed after the symbol is determined based on the scope of the current PC value.
 6. When watch expressions are registered from the [IOR panel](#) or the [CPU Register panel](#), the scope specification is automatically added.
 7. A watch expression "I" is recognized as a keyword for imaginary numbers. If you wish to specify "I" as a CPU register name, add ":REG" at the end.

<3> Register from other application

Select a character string of a variable of C language, CPU register, I/O register or assembler symbol from a external editor then do either one of the following.

- Drag and drop the target character string in this area in the desired watch panel (Watch1 to Watch4).
- Select the [Edit] menu >> [Paste] in this area in the desired watch panel (Watch1 to Watch4) after copying the target character string.


- Cautions 1.** Up to 128 watch expressions can be registered in one watch panel (if this restriction is violated, a message appears).
2. Due to compiler optimization, the data for the target variable may not be on the stack or in a register in blocks where that variable is not used. In this case, the target watch-expression value is displayed as "?".

- Remarks 1.** Each watch expression registered in each watch panel (Watch1 to Watch4) is managed in each panel and saved as the user information of the project.
2. More than one watch expression with the same name can be registered.

(d) Editing watch expression

To edit the registered watch expression, double-click the watch expression to be edited to change the watch expression to edit mode then directly edit from the keyboard (press the [Esc] key to cancel the edit mode). After editing the watch expression, press the [Enter] key to complete the editing.

(e) Deleting watch expression

To delete the registered watch expression, select the watch expression(s) to be deleted then click the  button on the toolbar or select [Delete] from the context menu.

(f) Setting of various events

Various events can be set to the selected watch expression by selecting [Access Break] or [Trace Output] from the context menu.

If an access event is set, the mark of the watch expression is changed (the event mark of a break event is displayed under the icon of the watch expression in layers).

When an event is set, the detailed information about the set event is reflected in the [Events panel](#).

Note that events are only set to the watch expressions that are global variables, static variables inside functions, or file-internal static variables.

See the following for details on how to set events.

- ["2.10.4 Stop the program with the access to variables/I/O registers"](#)
- ["2.13.4 Collecting an execution history only when conditions are met"](#)
- ["2.14.3 Measuring execution time in a section \[E1\] \[E20\]"](#)

(g) Jump to the address with memory definition

By selecting [Jump to Memory] from the context menu, the [Memory panel](#) (Memory1) opens with moving the caret to the address in which the selected watch expression is defined (if the Memory panel (Memory1) is already opened, jump to the panel).

Note that this operation is disabled when more than one watch expression is selected at the same time or the CPU register/I/O register is selected.

(2) [Value] area

The value of the registered watch-expression is displayed and changed (if the watch-expression is a function pointer, the function name is displayed in this area).

Notations and encodes can be selected by the button on the toolbar or the context menu item. In addition, a display format adding the value in hexadecimal number constantly can also be selected as well.

The default display format of the values is automatically decided depending on the type of the watch expression.


Table A-10. Display Format of Watch Expressions (Default)

Type of Watch Expression	Display Format
char, signed char, unsigned char	ASCII code with hexadecimal number
short, signed short, short int, signed short int, int, signed, signed int, long, signed long, long int, signed long int	Signed decimal number with hexadecimal number
unsigned short, unsigned short int, unsigned, unsigned int, unsigned long, unsigned long int	Unsigned decimal number with hexadecimal number
float, float _Complex, float _Imaginary	Float value (when size = 4 bytes)
double, long double, double _Complex, long double _Complex, double _Imaginary, long double _Imaginary	Double (when size = 8 bytes)
Pointers to char, signed char, unsigned char	Characters Encoding: ASCII
Pointers to other than char, signed char, unsigned char	Hexadecimal value
Arrays of char, signed char, unsigned char types	Characters Encoding: ASCII
bit, boolean, _boolean	Unsigned decimal value followed by a hexadecimal value written in ()
Enumeration type	Enumeration constant value followed by a hexadecimal value written in ()

Type of Watch Expression	Display Format
Label, address of immediate value, EQU symbol	Signed decimal value followed by a hexadecimal value written in ()
bit symbol	Unsigned decimal value followed by a hexadecimal value written in ()
Others	Hexadecimal value

Note Floating-point type values and complex type values are shown rounded to the nearest.

The meanings of the marks and colors displayed as the values of watch expressions are as follows (character colors and background colors depend on the configuration in the [\[General - Font and Color\]](#) category of the [Option dialog box](#)):

Display Example (Default)			Description
0x0	Character color	Blue	The value of the watch expression that the user is changing Press the [Enter] key to write to the target memory.
	Background color	Standard color	
0x0	Character color	Pink	The value of the watch expression that is displayed with the Real-time display update function
	Background color	Standard color	
0x0	Character color	Brown	The value of the watch expression that has been changed because of the execution of a program To reset the highlighting, select the  button on the toolbar or [Reset Color] from the context menu.
	Background color	Cream	
?	Character color	Gray	Variable that does not exist is registered as a watch expression or the value of the watch-expression cannot be retrieved (variable is out of the scope)
	Background color	Standard color	

- Remarks 1.** The I/O register that cause the microcontroller to operate when it is read is read-protected and therefore cannot be read. To read the value of read-protected I/O register, select [Force Read Value] from the context menu.
- 2.** Each watch expression acquires the value in the order it was registered.
As the timing to acquire a value is different, the values displayed may be different if the same I/O register is registered more than once.
- 3.** When a hexadecimal value is also given, then values in the specified notation and hexadecimal values are read separately. For this reason, the values with the specified notion and the hexadecimal values may differ due to the time lag between being read.

This area is provided with the following functions.

(a) Real-time display update function

Using the real-time display update function enables to display/modify the value of the watch expression not only while the program is stopped, but also in execution.

See "[\(4\) Displaying and changing memory contents during program execution](#)" for details on the real-time display update function.

(b) Changing values of watch expressions

To edit the value of the watch expression, change the value directly from the keyboard after double-clicking on the value to be edited (press the [Esc] key to cancel the edit mode).

After you edit the value of the watch expression, it is written to the target memory of the debug tool by pressing the [Enter] key, or moving the focus to outside the edit region.

See "(6) [Changing the contents of watch expressions](#)" for detail on how to change values of watch expressions.

(c) Saving the contents of watch expressions

By selecting the [File] menu >> [Save Watch Data As...], the [Save As dialog box](#) can be opened, and all the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).

See "(8) [Saving the displayed contents of watch expressions](#)" for details on the method for saving the contents of watch expressions.

(3) [Type (Byte Size)] area

The type information of watch expressions with the following format is displayed.

Watch Expression	Display Format	
Single CPU register	<Types of CPU register> (<Size ^{Note 1>})	
Single I/O register	<I/O register type> (<Access attribute> <Access type><Size ^{Note 1>})	
	Access attribute	R: Read only W: Write only R/W: Read/Write only
	Access type	1: Bit accessible 8: Byte accessible 16: Half word accessible 32: Word accessible
Unknown	?	
Others	<Watch-expression type that follow the C compiler's determination ^{Note 2>} (<Size ^{Note 1>})	

Notes 1. The size of the watch expression is displayed in bytes.

However, for bit I/O register or C language bit field, the size is displayed in bits and "bits" is added to the end of the number.

2. Types to be treated are displayed when compiling the watch expression.

3. The size of the watch expression is displayed in bytes.

In the case of double/long double type, type name is output in accordance with "Precision of the double type and long double type" in the [CPU] Category on the [Common Options] Tab in the CC-RX Property panel. The type name and size are output in float type when "Handles in single precision" is selected and in double type when "Handles in double precision" is selected.

(4) [Address] area

The address that each watch expression is mapped is displayed (hexadecimal number notation fixing).

If the watch expression is single CPU register or is unknown, "-" or "?" is displayed instead.

Remark When the watch expression is the bit I/O register, the bit-offset value is also displayed.

Example When the bit register is allocated to bit 4 of the address "0xFF40"
Display example:0xFF40.4

(5) [Memo] area

The user can write comments for the watch expressions/categories.

Each comment for a watch expression/category written in this area is saved individually as the user information of the project. Therefore, when any of the watch expression/category is deleted, the comment corresponding to it is also deleted.
















Note that when arrays or register are displayed expanded, the comment cannot be input for each element.

To edit the comment, input the character strings directly from the keyboard after double-clicking on the item to be edited (press the [Esc] key to cancel the edit mode).

Up to 256 character strings can be input (line feed code is ignored).

After editing the character strings, complete the editing by pressing the [Enter] key or moving the focus to outside the edit region.

[Toolbar]

	Reacquires all the values of the registered watch expression and updates the display. Note that read-protected I/O register values are not re-read.
	Resets highlighting of the selected watch expression whose value has been changed by executing a program. This item becomes invalid during execution of a program.
	Registers a new watch expression. Directly input the watch expression in the text box (see "(c) Registering new watch expression") Note that up to 128 watch expressions can be registered in one watch panel.
	Adds a new category (folder). Directly input the category name in the text box. Note that up to 64 categories can be created in one watch panel (categories cannot be created in categories).
	Deletes the selected character string(s). If the watch expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch expression is selected).
Notation	The following buttons to change the notation of a data value are displayed.
	Displays the value of the selected watch expression in the default notation (see "Table A-10. Display Format of Watch Expressions (Default)") according the type of variable (default).
	Displays the value of the selected item in hexadecimal number.
	Displays the value of the selected item in signed decimal number.
	Displays the value of the selected item in unsigned decimal number.
	Displays the value of the selected item in octal number.
	Displays the value of the selected item in binary number.
	Displays the value of the selected item in ASCII code.
	Displays the value of the selected item in Float. Note that this item becomes valid only when the selected watch-expression value is 4-byte data.
	Displays the value of the selected item in Double. Note that this item becomes valid only when the selected watch-expression value is 8-byte data.
	Adds the value in hexadecimal number enclosing with "("" at the end of the value of the selected item (except the item displayed in hexadecimal number).

[[File] menu (Watch panel-dedicated items)]

The following items are exclusive for the [File] menu in the Watch panel (other items are common to all the panels).
Note that all these items are invalid during execution of a program.

Save Watch Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(c) Saving the contents of watch expressions "). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Watch Data As...].
Save Watch Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(c) Saving the contents of watch expressions ").

[[Edit] menu (Watch panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Watch panel (other items are all invalid).

Cut	Deletes the selected character string(s) and copies them to the clipboard. If the watch expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch expression is selected).
Copy	Copies the contents of the selected range to the clipboard as character string(s). If the watch expression(s)/category(s) are selected, copies them to the clipboard (except when the expanded item of the watch expression is selected).
Paste	If texts are in editing, pastes the contents of the clipboard to the caret position. If texts are not in editing and the watch expression(s) are copied in the clipboard, registers them to the caret position.
Delete	Deletes the selected character string(s). If the watch expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch expression is selected).
Select All	If texts are in editing, selects all the character strings. If texts are not in editing, selects all the watch expressions/categories.
Rename	Renames the selected watch expression/category.
Find...	Opens the Find and Replace dialog box, with its [Find in Files] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Replace in Files] tab selected.

[Context menu]

Access Break	<p>This item becomes valid only when the selected watch expression is the global variable, the static variable inside functions, the file-internal static variable, or I/O register (multiple selections not allowed).</p> <p>The following cascade menus are displayed to set the access break event (see "(1) Set a break event (access type) to a variable/I/O register").</p>
Set Read Break to [Simulator]	Sets a break event with read access condition to the selected watch expression.
Set Write Break to [Simulator]	Sets a break event with write access condition to the selected watch expression.
Set R/W Break to [Simulator]	Sets a break event with read/write access condition to the selected watch expression.
Set Read Combination Break to [E1] [E20]	Sets a read-access break event in a selected watch expression as the condition for a combination break event.
Set Write Combination Break to [E1] [E20]	Sets a write-access break event in a selected watch expression as the condition for a combination break event.
Set R/W Combination Break to [E1] [E20]	Sets a read/write-access break event in a selected watch expression as the condition for a combination break event.
Trace Output	<p>This item becomes valid only when the selected watch-expression is a global variable, static variable inside functions, file-internal static variable, or I/O register (multiple selections not allowed).</p> <p>The following cascade menus are displayed to set the trace-related event (see "2.13.3 Collecting an execution history in a section" or "2.13.4 Collecting an execution history only when conditions are met").</p> <p>The corresponding Event mark is displayed at the top of the watch expression when an event is set.</p>
Record Reading Value	Sets a Point Trace event to record the values in the trace memory when the selected watch expression is accessed for read (see "(1) When an access to a variable or I/O register occurred ").
Record Writing Value	Sets a Point Trace event to record the values in the trace memory when the selected watch expression is accessed for write (see "(1) When an access to a variable or I/O register occurred ").
Record R/W Value	Sets a Point Trace event to record the values in the trace memory when the selected watch expression is accessed for read/write (see "(1) When an access to a variable or I/O register occurred ").
Record Start R/W Value	Sets a trace event that causes trace recording to start upon read/write access to a selected watch expression (see "(1) Setting a trace start event and a trace end event ").
Record End R/W Value	Sets a trace event that causes trace recording to end upon read/write access to a selected watch expression (see "(1) Setting a trace start event and a trace end event ").
Trace	Opens the Trace panel and displays the acquired trace data.

Timer Settings [E1] [E20]	<p>This item is enabled only when the selected watch expression is a global variable, a static variable in function, a static variable in file, or an I/O register (plural selections not accepted).</p> <p>Displays the following cascaded menu to set timer-related events (see Section “2.14.3 Measuring execution time in a section [E1] [E20]”).</p> <p>After an event is set, an event mark is displayed at the beginning of the watch expression concerned.</p>
Set Timer Start R/W Value	Sets an event that causes the timer to start upon read/write access to a selected watch expression (see “ (a) How to set a timer start event ”).
Set Timer <N>	Specify a channel ^{Note} in which a timer start event is set.
Set Timer End R/W Value	Sets an event that causes the timer to finish upon read/write access to a selected watch expression (see “ (b) How to set a timer end event ”).
Set Timer <N>	Specify a channel ^{Note} in which a timer start event is set.
Periodic Updating	The following cascade menus are displayed to set for the real-time display update function (see “ (a) Real-time display update function ”).
Periodic Updating Options	Opens the Property panel to set for the real-time display update function.
Refresh	<p>Reacquires all the values of the registered watch expression and updates the display.</p> <p>Note that the values of read-protected I/O register are not re-read.</p>
Force Read Value	<p>Forcibly reads once the values of the read-protected I/O register.</p> <p>This item becomes invalid during execution of a program.</p>
Add New Watch	<p>Registers a new watch expression. Directly input the watch-expression in the text box (see “(c) Registering new watch expression”).</p> <p>Note that up to 128 watch expressions can be registered in one watch panel.</p>
Create Category	<p>Adds a new category (folder). Directly input the category name in the text box.</p> <p>Note that up to 64 categories can be created in one watch panel (categories cannot be created in categories).</p>
Delete	<p>Deletes the selected character string(s).</p> <p>If the watch expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch expression is selected).</p>
Cut	<p>Deletes the selected character string(s) and copies them to the clipboard.</p> <p>If the watch expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch expression is selected).</p>
Copy	<p>Copies the contents of the selected range to the clipboard as character string(s).</p> <p>If the watch expression(s)/category(s) are selected, copies them to the clipboard (except when the expanded item of the watch expression is selected).</p>
Paste	<p>If texts are in editing, pastes the contents of the clipboard to the caret position.</p> <p>If texts are not in editing and the watch expression(s) are copied in the clipboard, registers them to the caret position.</p>

Notation	The following cascade menus are displayed to specify the notation.
AutoSelect	Displays the value of the selected watch-expression in the default notation (see " Table A-10. Display Format of Watch Expressions (Default) ") according the type of variable (default).
Hexadecimal number	Displays the value of the selected item in hexadecimal number.
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned decimal number	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.
Include Hexadecimal Value	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item (except the item displayed in hexadecimal number).
Float	Displays the value of the selected item in Float. Note that when the selected watch expression value is not 4-byte data, or has the type information, displays it in the default notation (see " Table A-10. Display Format of Watch Expressions (Default) ").
Double	Displays the value of the selected item in Double. Note that when the selected watch-expression value is not 8-byte data, or has the type information, displays it in the default notation (see " Table A-10. Display Format of Watch Expressions (Default) ").
Decimal Notation for Array Index	Displays array indexes on this panel in decimal number (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel in hexadecimal number.
Encoding	The following cascade menus are displayed to specify the character code.
ASCII	Displays the value of the selected item in ASCII code (default).
Shift_JIS	Displays the value of the selected item in Shift_JIS code.
EUC-JP	Displays the value of the selected item in EUC-JP code.
UTF-8	Displays the value of the selected item in UTF-8 code.
UTF-16	Displays the value of the selected item in UTF-16 code.
Size Notation	The following cascade menus are displayed to specify the size notation.
1 Bytes	Displays the value of the selected item as 8-bit data.
2 Bytes	Displays the value of the selected item as 16-bit data.
4 Bytes	Displays the value of the selected item as 32-bit data.
8 Bytes	Displays the value of the selected item as 64-bit data.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the address which the selected watch expression is defined (see " (g) Jump to the address with memory definition ").
Reset Color	Resets highlighting of the selected watch expression whose value has been changed by executing a program. This item becomes invalid during execution of a program.

Note The specifiable number of channels differs between the RX600 and RX200 Series, as shown below.

RX600 Series: 2 (32 bits * 2) or 1 (64 bits * 1)

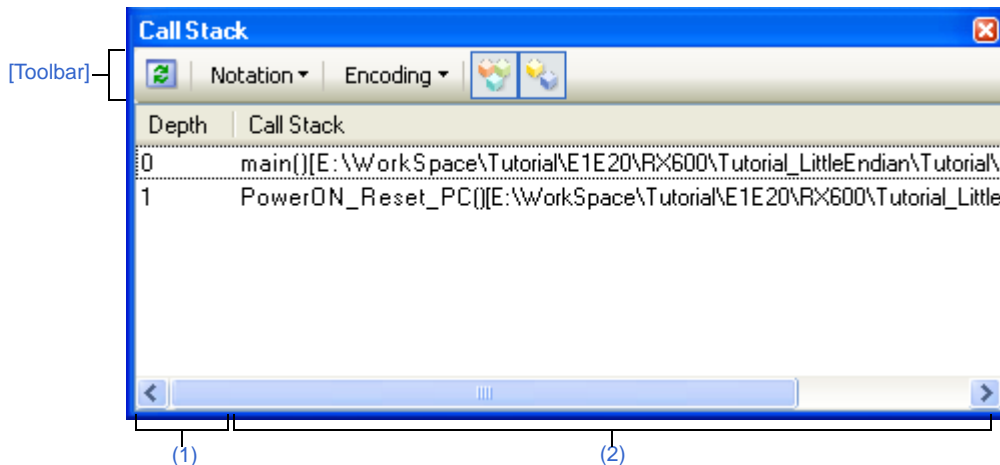
RX200 Series: 1 (24 bits * 1)

Call Stack panel

This panel displays call-stack information for function calls (see “2.12.1 Display call stack information”).
This panel can only be opened when CubeSuite+ is connected with the debug tool.

Caution This panel is left blank while the program is in execution.
Each area on this panel are displayed at the time the program has stopped running.

Figure A-32. Call Stack Panel



Here, the following items are explained.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] Menu (Call Stack Panel-Only Items)]
- [[Edit] Menu (Call Stack Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Call Stack] from the [View] menu.



[Description of each area]**(1) [Depth] area**

Displays the depth of calls.

The callers of functions are assigned numbers in order starting from 1, with the line showing the current PC position assigned a 0.

(2) [Call Stack] area

Displays the current source position and call-stack information about the calls placed on the stack (e.g., function caller position and parameters to each function).

The form in which information is displayed in this area differs depending on the status of which toolbar button,  or , is selected, and which item on context menu, [Show Parameter] or [Show Module File Name], is selected, as shown below.

Condition	Display Format
- Show parameters - Show module file names	<code><function>(<parameter>=<parameter value^{Note}>, ...)[<module file name>\$<file name>#<line number>]</code> (default)
- Show parameters - Do not show module file names	<code><function>(<parameter>=<parameter value^{Note}>, ...)[<file Name>#<line number>]</code>
- Do not show parameters - Show module file names	<code><function>()[<module file name>\$<file name>#<line number>]</code>
- Do not show parameters - Do not show module file names	<code><function>()[<file name>#<line number>]</code>

Note If the parameter value is a character string, up to 20 characters are displayed.

Remark An array of parameters are passed as a pointer, not as an array (C language specifications). Therefore, if parameters are comprised of an array, they are handled as pointer when displayed.

This area has the following features:

(a) Jump to the source line/ disassemble

Selecting [Jump to Source] from the context menu will open the [Editor panel](#), with the caret moved to the source line from which the function at the current caret position is called. (The view will jump to the Editor panel if it is already open.)

Similarly, selecting [Jump to Disassemble] will open the [Disassemble panel](#) (Disassemble1), with the caret moved to the address from which the function at the current caret position is called. (The view will jump to the Disassemble panel (Disassemble1) if it is already open.)

Remark Double-clicking a line will also make you jump to the corresponding source line.














(b) Saving of call-stack information

Selecting [Save Call Stack Data As ...] from the [File] menu will open the [Save As dialog box](#) in which you can save all contents of this panel to a text file (*.txt) or CSV file (*.csv).

For details on how to save call-stack information, see "(4) [Saving the displayed contents of call-stack information](#)".

[Toolbar]

Each button in the toolbar are disabled during program execution.

	Obtains latest information from the debug tool and updates display.
Notation	Shows the following buttons that change the form in which values are displayed.
	Displays values on this panel in per-variable predetermined notation (default).
	Displays values on this panel in hexadecimal.
	Displays values on this panel in decimal.
	Displays values on this panel in octal..
	Displays values on this panel in binary.
Encoding	Shows the following buttons that change the encode in which string variables are displayed.
	Displays string variables in ASCII code (default).
	Displays string variables in Shift_JIS code.
	Displays string variables in EUC-JP code.
	Displays string variables in UTF-8 code.
	Displays string variables in UTF-16 code.
	Adds a module file name to the information displayed (default).
	Adds function call parameters to the information displayed (default).

[[File] Menu (Call Stack Panel-Only Items)]

The [File] menu used exclusively for the call-stack panel is as follows. (The other items are shared.)

However, all of these items are disabled during program execution.

Save Call Stack Data	Saves the contents of this panel to a text file (*.txt) or CSV file (*.csv) that has been saved previously (see "(b) Saving of call-stack information"). If this item is selected for the first time after startup, the same operation as [Save Call Stack Data As ...] would have been selected is performed.
Save Call Stack Data As...	Opens the Save As dialog box in order to save the contents of this panel to a specified text file (*.txt) or CSV file (*.csv) (see "(b) Saving of call-stack information").

[[Edit] Menu (Call Stack Panel-Only Items)]

The [Edit] menu used exclusively for the call-stack panel is as follows. (All other items are disabled.)

Copy	Copies the content of a selected line as character string to the clipboard.
Select All	Puts the item into all selected state.
Find...	Opens the Find and Replace dialog box, with its [Find in Files] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Replace in Files] tab selected.

[Context menu]

Each item on the context menu are disabled during program execution.

Copy	Copies the content of a selected line as character string to the clipboard.
Show Module File Name	Adds a module file name to the line when it is displayed (default).
Show Parameter	Adds function call parameters to the line when it is displayed (default)
Notation	Shows the following cascaded menu to specify the form in which values are displayed.
AutoSelect	Displays values on this panel in per-variable predetermined notation (default).
Hexadecimal	Displays values on this panel in hexadecimal.
Decimal	Displays values on this panel in decimal.
Octal	Displays values on this panel in octal.
Binary	Displays values on this panel in binary.
Encoding	Shows the following cascaded menu to specify the encode in which values are displayed.
ASCII	Displays string variables in ASCII code (default).
Shift_JIS	Displays string variables in Shift_JIS code.
EUC-JP	Displays string variables in EUC-JP code.
UTF-8	Displays string variables in UTF-8 code.
UTF-16	Displays string variables in UTF-16 code.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1), with the caret on it moved to the address from which the function indicated by a selected line is called.
Jump to Source	Opens the Editor panel , with the caret on it moved to the source line from which the function indicated by a selected line is called.
Jump to Local Variable at This Time	Opens the Local Variables panel that displays local variables for the function indicated by a selected line.

Trace panel

This panel is used to display trace data recording the execution history of the program (see ["2.13 Collecting an Execution History"](#)).

The trace data displays by mixing the disassembled text and source text by default, but it is also possible to display either one of these by selecting the [Display mode](#).

After the execution of the program is stopped, the display position is automatically updated such that the latest trace data is displayed.

This panel appears only when connected to the debug tool.

Caution [E20(JTAG) [RX600 Series]]

Part of the trace functions and Real-time RAM Monitor (RRM) functions can be used only on a mutually exclusive basis.

Remark When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A-33. Trace Panel [E1] [E20]

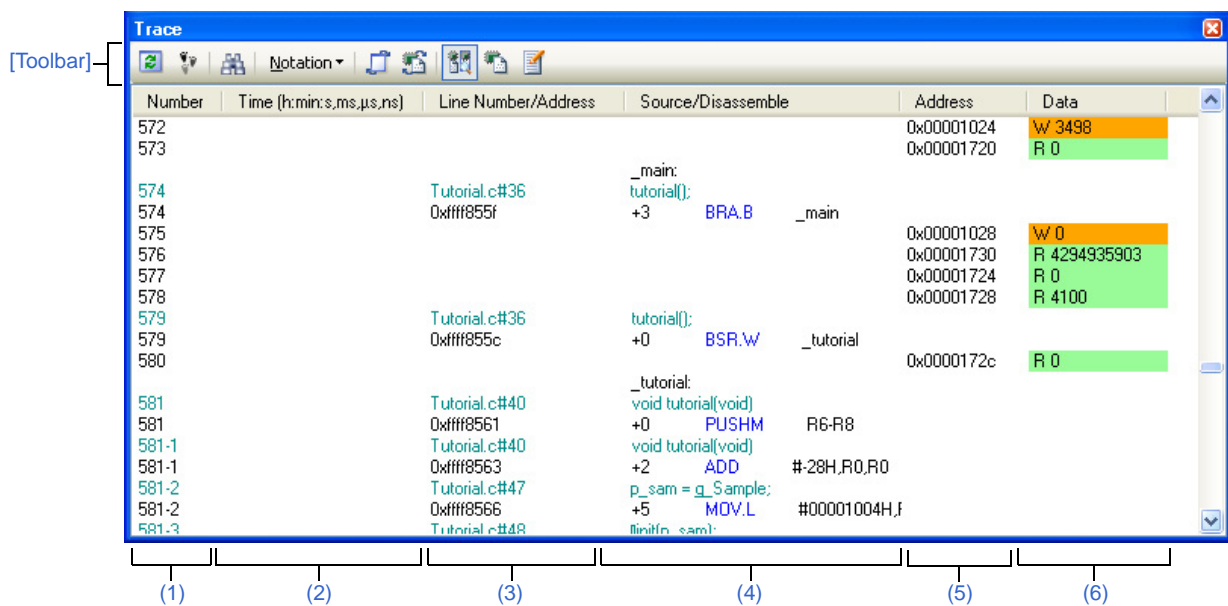
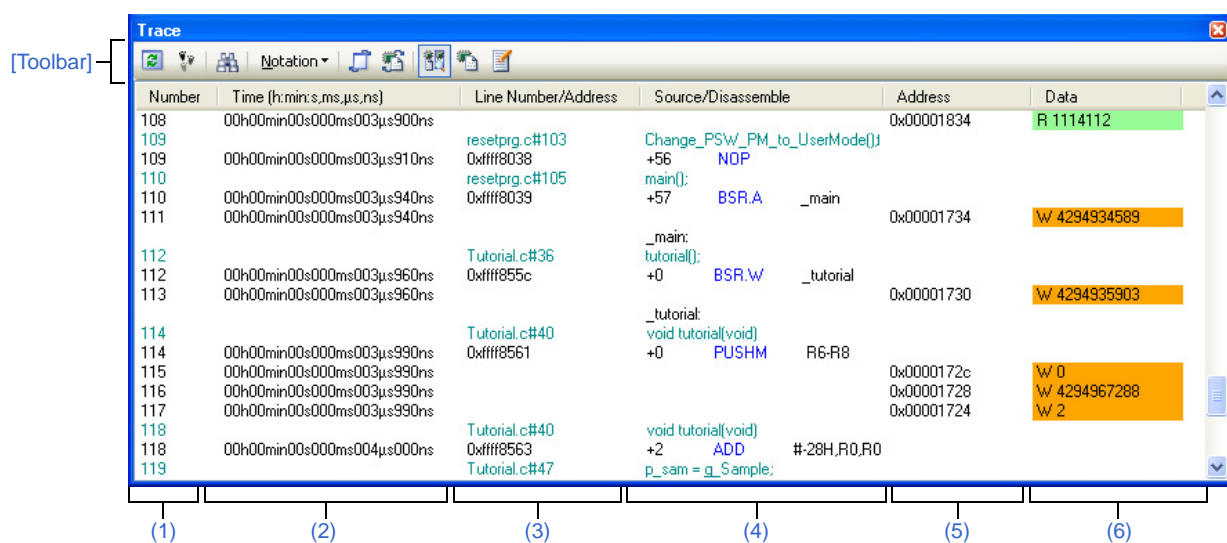


Figure A-34. Trace Panel [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Trace panel-dedicated items)]
- [[Edit] menu (Trace panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Trace].
- On the [Editor panel/Disassemble panel](#), select [Trace Settings] >> [Show Trace Result] from the context menu.

[Description of each area]**(1) [Number] area**

The trace number showing the trace frame is displayed.

(2) [Time (h:min:s,ms,μs,ns)] area

This area displays the time required from the execution start of the program to the execution start of an instruction of each frame or generation of memory access cause.

The time is displayed in units of "hours, minutes, seconds, milliseconds, microseconds and nanoseconds".

If overflow occurs, this area is displayed in invalid color (gray)

Remarks 1. [Simulator]

The question of whether to set the time display as an integrated value or differential value depends on the setting of the [Accumulate trace time] property on the [Trace] category on the [\[Debug Tool Settings\] tab](#) of the [Property panel](#).

2. [E1] [E20]

Time display function is not supported.

(3) [Line/Address] area

The address of the assemble code or the line number of a source file is displayed.

The notation of a data value can be selected by the button on the toolbar or the context menu item.

The display formats are as follows:

Type of Display Line	Display Format
Instruction (disassemble results)	<Address>
Source text	<File name>#<Line number>
Label	-
Point trace results	-

Remark Since the following execution histories are not displayed, the line numbers displayed are not consecutive numbers.

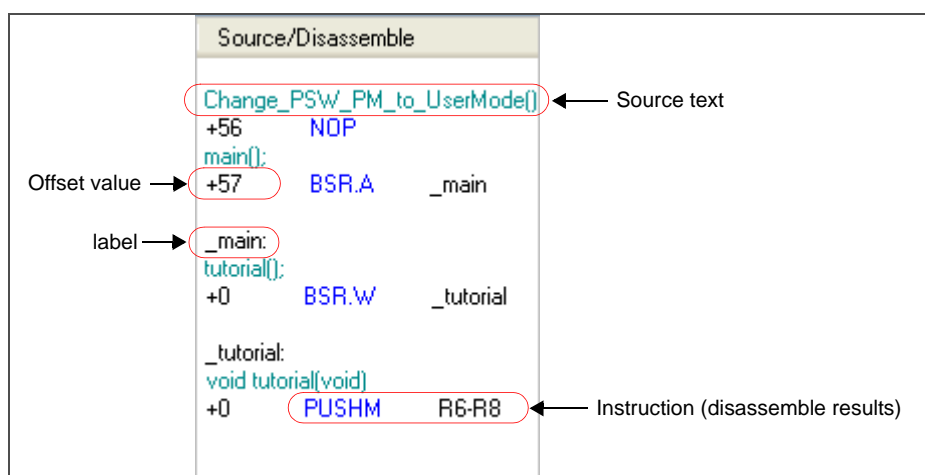
- CPU register access
- Invalid fetch

(4) [Source/Disassemble] area

The collected trace data is displayed as follows.

The items displayed in this area differ depending on the selection of the display mode (see "(a) [Display mode](#)").

Figure A-35. Display Contents of the [Source/Disassemble] Area (Default)



Label	The label is displayed when a label is defined for the address.
Offset value	The offset value from the nearest label is displayed when a label is defined for the address.
Source text	The corresponding source text is displayed when the Mixed display mode or Source display mode is selected. However, when a place where no debugging information is present is executed, "<No Debug Information>" is displayed.
Instruction (disassemble results)	The corresponding instructions are displayed as the result of disassembling when the Mixed display mode or Disassemble display mode is selected. The mnemonics are shown highlighted. Note

Note At a frame for which not all the trace data was fetched, "(LOST)" is displayed. In this case, the corresponding line is shown in error color (the error color depends on the configuration in the [\[General - Font and Color\]](#) category of the [Option dialog box](#)).

This area is provided with the following functions.

(a) Display mode

It is possible to select the following three display modes by selection of a button on the toolbar or the context menu.


Display Mode	Display Contents
Mixed display mode	Displays the instruction (disassemble results), labels, source text (corresponding source line) (default).
Disassemble display mode	Displays the instruction (disassemble results), labels.
Source display mode	Displays the source text (corresponding source line) and break causes. However, when a place where no debugging information is present is executed, "<No Debug Information>" is displayed.

(b) Jumping to source line or disassemble

By selecting [Jump to Source] from the context menu, the [Editor panel](#) opens with moving the caret to the source line corresponding to the line at the current caret position (if the Editor panel is already opened, jump to the panel).

In addition, similarly by selecting [Jump to Disassemble], the [Disassemble panel](#) (Disasemmble1) is opened with moving the caret to the address corresponding to the fetch address of the line at the current caret position (if the Disassemble panel is already opened, jump to the panel (Disassemble1)).

(c) Linking with other panels

By clicking the  button on the toolbar, or selecting [Window Connecting] >> [Connect Source Window]/[Connect Disassemble Window] from the context menu, it is possible to link and display the corresponding places on the [Editor panel/Disassemble panel](#), with the address of the caret position on this panel used as the pointer (no movement of the focus is done).

(d) Pop-up display

By superimposing the mouse cursor on a line, all the area (item) data corresponding to that line is displayed as a pop-up in tandem shape.

(e) Saving trace data

The [Data Save dialog box](#) can be opened by selecting the [File] menu >> [Save Trace Data As...], and the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).

See "[2.13.8 Saving the displayed content of an execution history](#)" for details on the method for saving trace data.

(5) [Address] area

The target address of memory access is displayed.

However, in the event of access to I/O register, the I/O register name is displayed instead of the address (when a plurality is accessed these are displayed in the following lines).

The radix of a data value can be selected by the button on the toolbar or the context menu item.

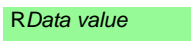

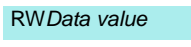
(6) [Data] area

The accessed data value and the access type at that time are displayed.













However, CPU register access is not displayed.

The notation of a data value can be selected by the button on the toolbar or the context menu item.

The display format of the data value and the access type are as follows (character colors and background colors depend on the configuration in the [\[General - Font and Color\] category](#) of the [Option dialog box](#)):

Display Example (Default)			Memory Access Type
	Character color	Standard color	Read access
	Background color	Palegreen	
	Character color	Standard color	Write access
	Background color	Orange	
	Character color	Standard color	Read and write access
	Background color	Paleturquoise	

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel. This item becomes invalid during execution of a program.
	Clears the trace memory and the display of this panel (initialized). This item becomes invalid during execution of a program.
	Opens the Trace Search dialog box .
Notation	The following buttons to change the notation of a data value are displayed. This item becomes invalid during execution of a program.
	Displays values on this panel in hexadecimal number (default).
	Displays values on this panel in decimal number.
	Displays values on this panel in octal number.
	Displays values on this panel in binary number.
	Links with the Editor panel .
	Links with the Disassemble panel .
	Sets to the Mixed display mode as the display mode (default). This item becomes invalid during execution of a program.
	Sets to the Disassemble display mode as the display mode. This item becomes invalid during execution of a program.
	Sets to the Source display mode as the display mode. This item becomes invalid during execution of a program.

[[File] menu (Trace panel-dedicated items)]

The following items are exclusive for the [File] menu in the Trace panel (other items are common to all the panels).
Note that all these items are invalid during execution of a program.

Save Trace Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(e) Saving trace data "). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Trace Data As...].
Save Trace Data As...	Opens the Data Save dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(e) Saving trace data ").

[[Edit] menu (Trace panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Trace panel (other items are all invalid).
Note that all these items are invalid during execution of a program.

Copy	Copies the contents of the selected line to the clipboard (multiple line selections impossible).
Find...	Opens the Trace Search dialog box .

[Context menu]

Clear Trace	Clears the trace memory and the display of this panel (initialized). This item becomes invalid during execution of a program.
Find...	Opens the Trace Search dialog box . This item becomes invalid during execution of a program.
Copy	Copies the contents of the selected line to the clipboard (multiple line selections impossible). This item becomes invalid during execution of a program.
Mixed Display	Sets to the Mixed display mode as the display mode. This item becomes invalid during execution of a program.
Disassemble View	Sets to the Disassemble display mode as the display mode. This item becomes invalid during execution of a program.
Source View	Sets to the Source display mode as the display mode. This item becomes invalid during execution of a program.
Notation	The following cascade menus are displayed to specify the notation. This item becomes invalid during execution of a program.
Hexadecimal number	Displays values on this panel in hexadecimal number (default).
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.
Window Connecting	The following cascade menus are displayed to link with other panels.
Connect Source Window	Links with the Editor panel .
Connect Disassemble Window	Links with the Disassemble panel .
Jump to Disassemble	Opens the Disassemble panel (Disassemble1) and jumps to the fetch address corresponding to the selected line in this panel.
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the selected line in this panel.

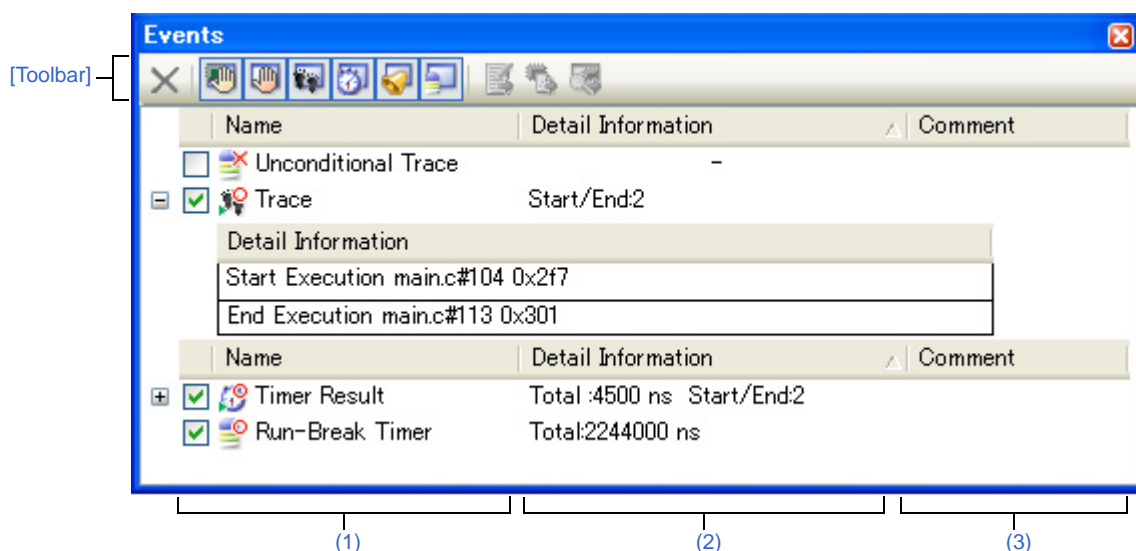
Events panel

This panel is used to display the detailed information about the events that are set on the [Editor panel/Disassemble panel/Watch panel](#). On this panel, you can change the setting state of the event between valid/invalid and delete the event (see "2.17 Event Management").

This panel appears only when connected to the debug tool.

- Remarks 1.** See "2.17.7 Points to note regarding event setting" for more information about the setting of events.
2. This panel also manage the events that are set on the Function panel or Variable panel of the analyze tool (Program Analyzer).
 3. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A-36. Events Panel



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[\[Edit\] menu \(Events panel-dedicated items\)\]](#)
- [\[Context menu\]](#)

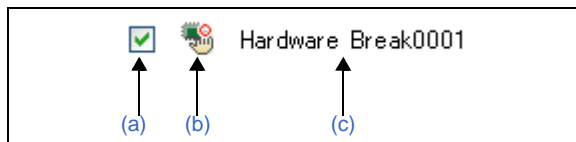
[How to open]

- From the [View] menu, select [Event].
- On the [Editor panel](#) or [Disassemble panel](#), select [Timer Settings] >> [View Result of Timer] from the context menu.
- On the [Editor panel](#) or [Disassemble panel](#), move the caret to an event mark and then select [View Details in Event Panel] on context menu.

[Description of each area]

(1) [Name] area

A list of the event names that have currently been set is displayed in the following format.



Remark It is possible to limit the event to be displayed by clicking the button on the toolbar (see "[Toolbar]").

(a) Check box

The setting state of the event is displayed/changed.

Note that the [Event mark](#) is changed depending on the setting state of the event.

<input checked="" type="checkbox"/>	Enabled	Event occurs when the specified condition is met. It is possible to set the event to an invalid state by removing the check.
<input type="checkbox"/>	Disabled	Event does not occur when the specified condition is met. It is possible to set the event to a valid state by removing the check.
<input type="checkbox"/>	Pending	The conditions that have been specified cannot be set with the program of the debugging target. It is not possible to operate the check box.

- Remarks 1.** It is not possible to set the Run-Break Timer event to an invalid/suspended state.
- 2.** The setting of the Unconditional Trace event and the Trace event to valid or invalid state is exclusively controlled. Therefore, the Unconditional Trace event, which is a built-in event, is valid state by default, but if either a Start Tracing event/Stop Tracing event is set, it automatically becomes invalid state, and the Trace event, which is a event name that is collectively called with a Start Tracing event and a Stop Tracing event, becomes valid state. Conversely, if the set Trace event is invalid state, the Unconditional Trace event automatically becomes valid state.

(b) Event mark

The event mark shows the type of event, and in addition shows the current setting state.

The meanings of the marks displayed are as follows:

Table A-11. Event Mark

Event Type	Valid State	Invalid State	Suspended State	Note
Hardware Break				-
Software Break [E1] [E20]				-
Combination break [E1] [E20]				-
Unconditional Trace			None	-
Run-Break Timer		None	None	-
Trace				Used only in the Events panel
Trace start				Used only in the Editor panel / Disassemble panel
Trace end				
Timer Result [E1] [E20]				Used only in the Events panel
Timer start				Used only in the Editor panel / Disassemble panel
Timer end				
Point Trace				-
Printf event (Action event)				-
Interrupt event (Action event) [Simulator]				-
Setting of two or more events	Note 1	Note 2	Note 3	Used only in the Editor panel / Disassemble panel

Notes 1. There is one or more event with valid state.

2. There is no event with valid state and at least one event with invalid state.

3. All the set events are suspended state

(c) Event name

The event type and ID number are displayed as the event name.

A number from 0001 is automatically provided as the ID number for each event (no renumbering of the ID number is done even in the event that an event that has been set once is deleted).

Event types that are displayed are as follows:

Table A-12. Event Type

Event Type	Description
Hardware Break (Break ^{Note1})	Breaks the program when the condition is met while the debug tool monitors the break condition all the time during program execution. -> See "2.10.2 Stop the program at the arbitrary position (breakpoint)" -> See "2.10.4 Stop the program with the access to variables/I/O registers"
Software Break (Break ^{Note1}) [E1] [E20]	Breaks the program when the instruction, which an address code to break is rewritten for the break instruction, is executed. -> See "2.10.2 Stop the program at the arbitrary position (breakpoint)"
Combination break [E1] [E20]	When, while the debug tool successively is checking plural break conditions during program execution, the combination condition is met, this event causes the program to break. -> See "2.10.5 Set multiple break events in combination (Combination break) [E1] [E20]"
Unconditional Trace	Automatically collects the trace data with start of a program execution, and stops collecting the trace data with stop of the program execution. This event cannot be deleted because of the built-in event ^{Note2} (this event is Enabled by default). -> See "2.13.2 Collecting an execution history up to a halt"
Run-Break Timer	Automatically measures the execution time of a program with start of the program execution, and stops the measurement with stop of the program execution. This event cannot be deleted because of the built-in event ^{Note 2} (this event is Enabled by default). -> See "2.14.2 Measuring execution time from start to stop"
Trace	Starts/stops collecting the trace data when the condition specified with "Trace start" and "Trace end" is met (this event is displayed when a Start Tracing event or a Stop Tracing event is set). -> See "2.13.3 Collecting an execution history in a section"
Timer Result ^{Note3} [E1] [E20]	Starts/stops measuring the execution time of a program when the condition specified with "Timer start" and "Timer end" is met (this event is displayed when a Start Timer event or a Stop Timer event is set). -> See "2.14.3 Measuring execution time in a section [E1] [E20]"
Point Trace	Records the information as the trace data when accessing the specified variable or I/O register during execution of a program. -> See "2.13.4 Collecting an execution history only when conditions are met"
Printf	Executes printf command in software processing after temporary stopping a program in execution at an arbitrary position (action event). -> See "2.16.1 Insert printf"
Interrupt [Simulator]	This event generates an interrupt at any place during program execution (action event). -> See "2.16.2 Insert an interrupt event [Simulator]"

- Notes**
1. A breakpoint that is set by a one click operation of the mouse is displayed "Break" (see "(2) Set a breakpoint").
 2. This is set in the debug tool by default.
 3. **[E1] [E20]**
A channel number indicating the range to be measured by the timer is given to this event type (e.g. Timer Result 1). The number of measurement ranges varies with the Microcontroller.
 - RX600 Series: 2 ranges
 - RX200 Series: 1range

Remark The following event types are also displayed when an event (breakpoint or break event) is set on the Function panel or the Variable panel of the analyze tool (Program Analyzer)

- In the case of a breakpoint at a function: "Break at start of function"
- In the case of a break event to a variable: "Access break to variable"

(2) [Detail Information] area

Detailed information about each event is displayed.

The contents of the information that is displayed differ depending on the event type as follows:

Table A-13. Detailed Information with Event Type

Event Type	Display Contents ^{Note1}		
Hardware Break (Generation condition: Execution-related)	Format1	<Generation condition> <File name#Line number> <Address>	
	Example	Before Execution	main.c#39 0x100
		Before Execution	- 0x300
		Execution	main.c#39 0x300 [Simulator]
	Format2	<Generation condition> <Symbol + Offset> <Address>	
	Example	Before Execution	funcA + 0x10 0x100
		Before Execution	- 0x300
Hardware Break (Generation condition: Access-related)	Format1	<Generation condition> <File name# Variable name> <Address(range)> <Comparison condition> <Comparison value>	
	Example	Read	main.c#variable1 0x100 - 0x101 == 0x5
		Write	sub.c#variable2 0x200 - 0x200 == 0x7
		Read/Write	sub2.c#variable3 0x300 - 0x303 == 0x8
	Format2	<Generation condition> <File name#Function name# Variable name> <Address(range)> <Comparison condition> <Comparison value>	
	Example	Read	main.c#func1#variable1 0x100 - 0x101 == 0x10
	Format3	<Generation condition> <Variable name> <Address(range)> <Comparison condition> <Comparison value>	
	Example	Write	variable1 0x100 - 0x101 == 0x10
Software Break [E1] [E20]	Format1	<Generation condition> <File name#Line number> <Address>	
	Example	Before Execution	main.c#40 0x102
		Before Execution	sub.c#101 0x204
	Format2	<Generation condition> <Symbol + Offset> <Address>	
	Example	Before Execution	funcA + 0x12 0x102

Event Type	Display Contents ^{Note1}	
Combination break (Generation condition: Execution-related, Access- related) [E1] [E20]	Format	<Combination condition> <Detailed information of combination break>
	Example	OR - After execution main.c#100 0x300 - After execution funcA + 0x10 0x100 - Write sub.c#variable2 0x200 - 0x200 == 0x7 - Read/Write sub2.c#variable3 0x300 - 0x303 ==0x8
Unconditional Trace	Format	-
	Example	-
Run-Break Timer	Format	Total: <Total execution time>
	Example	Total: 1000ms
		Total: OVERFLOW
	Format2	Total:<Total execution time> Execution Cycle Count:<Total execution cycle count> Execution Instruction Count:<Total execution instruction count> [Simulator]
Trace (Generation condition: Execution-related, Access- related) [E1] [E20]	Format	Start: <Combination condition for trace start> End: <Combination condition for trace end> Total number of starts and ends: <Total number of trace start/trace end events> ^{Note2} <Start/End> <Trace start/trace end detailed information>
	Example	Start: OR End: OR Total of Start/End: 6 - Start After Execution main.c#100 0x300 - Start After Execution funcA + 0x100 0x300 - Start Write variable1 0x100-0x101==0x10 - End After Execution main.c#200 0x100 - End After Execution funcA + 0x10 0x100 - End Read main.c#variable1 0x100-0x101==0x5
Trace (Generation condition: Execution-related, Access- related) [Simulator]	Format	Start: OR End: OR T Total number of starts and ends: <Total number of trace start/ trace end events> ^{Note2} <Start/end><Trace start/trace end detailed information>
	Example	Start: OR End: OR Total of Start/End: 6 - Start Execution main.c#100 0x300 - Start Execution funcA + 0x100 0x300 - Start Write variable1 0x100-0x101==0x10 - End Execution main.c#200 0x100 - End Execution funcA + 0x10 0x100 - End Read main.c#variable1 0x100-0x101==0x5

Event Type	Display Contents Note1	
Timer Result (Generation condition: Execution-related) [E1] [E20]	Format	Total:<Total execution time > Total of Start/End: <Total number of Start timer/Stop timer> Note 2 - <Total execution time> <Pass Count> <Average> <Max> <Min> - <Start/End> <Detailed information of Start timer/Stop timer>
	Example	Total: 10ms Total of Start/End: 4 - Total: 10ms Pass Count: 5 Average: 2ms Max: 4ms Min: 1ms - Start After Execution main.c#100 0x300 - Start After Execution funcA + 0x30 0x100 - End After Execution main.c#100 0x300 - End After Execution funcA + 0x50 0x100
Point Trace (Generation condition: Access-related)	Format1	<Generation condition> <Variable name> <Variable address>
	Example	Read variable1 0x100
	Format2	<Generation condition> <File name# Variable name> <Variable address>
	Example	Write sub.c#variable2 0x200
	Format3	<Generation condition> <File name#Function name# Variable name> <Variable address>
	Example	Read/Write sub.c#func1#variabl3 0x300
Printf event (Action event)	Format	<Generation condition> <File name#Line number> <Address> <Setting of Printf event>
	Example	Before Execution main.c#39 0x100 aaa, bbb, ccc
		After Execution sub.c#100 0x200 Result of aaa : aaa
Interrupt event (Action event) [Simulator]	Format	<Generation condition><File name # line number><Address> interrupt vector: <Interrupt vector>Priority level: <Interrupt priority>
	Example	Before execution main.c#39 0x100 Interrupt vector: 1c Priority level: 7

Notes 1. Following are the details on the display format.

<Generation condition>	Displays one of the following conditions. [E1] [E20] Execution: Before Execution or After Execution Access: Read, Write, Read/Write [Simulator] Execution: Execution Access: Read, Write, Read/Write
------------------------	--

<File name # Line number>	Shows the source file name and the line number in the source file. Display format is the same as the watch type scope specification expression. When multiple load module files are downloaded, <Load module file name\$File name#Line number> is displayed. For those events set in the Disassemble panel , displays <Line number> in the format <Symbol + offset> in the condition below. - Line information exists and the specified position where the event is set is not the top of the line information - Line information does not exist and symbol information exists. Shows <Line number> in "-" in the following condition. - Line information and symbol information do not exist.
<Variable name>	Shows the variable name in the source file. Display format is the same as the watch type scope specification expression.
<Comparison condition>	Condition to compare (==) is shown. If the comparison value is not specified, comparison condition is not shown.
<Comparison value>	Comparison value is shown. If the comparison value is not specified, comparison condition is not shown.
<Address>	Address in the memory area is shown (only in hex number).
<Combination condition>	Displays one of the following conditions. OR, AND, Sequential
<detailed information of combination break>	Detailed information about combination break event is displayed.
<Total>	Shows the measurement result of the timer total execution time. The unit is either of ns/μs/ms/s/min (if, however, the unit is in "min", a value in "s" unit also appears). If a timer overflow occurs (see " 2.14.4 Range of measurable time "), or if the illegal value was acquired, "OVERFLOW" is displayed.
<Combination condition for trace start>	Displays one of the following conditions. OR, AND, Sequential
<Combination condition for trace end>	Displays OR condition.
<Total number of trace start/trace end event>	The total number of trace start events and trace end events is displayed.
<Start/End>	Shows whether the contents of the detailed information is start event or the stop event.
<Trace start/trace end detailed information>	Detailed information about trace start/trace end event is displayed.
<Pass Count>	Shows the measurement result of the pass count of the timer. If a timer overflow occurs (see " 2.14.4 Range of measurable time "), or if the illegal value was acquired, "OVERFLOW" is displayed.
<Average>	Shows the measurement result of average execution of the timer. The unit is either of ns/μs/ms/s/min (if, however, the unit is in "min", a value in "s" unit also appears).

<Max>	Shows the measurement result of the maximum execution time of the timer. The unit is either of ns/ μ s/ms/s/min (if, however, the unit is in "min", a value in "s" unit also appears).
<Min>	Shows the measurement result of the minimum execution time of the timer. The unit is either of ns/ μ s/ms/s/min (if, however, the unit is in "min", a value in "s" unit also appears).
<Total number of cycles executed>	Displays the total number of cycles executed between Run-Break as the result of measurements made.
<Total number of instructions executed>	Displays the total number of instructions executed between Run-Break as the result of measurements made.
<Set print event>	Shows the variable expression and the character strings specified in the Action Events dialog box .
<Interrupt vector>	Displays the interrupt vector specified in the Action Events dialog box or Detailed Settings of Interrupt Events dialog box [Simulator] .
<Interrupt priority>	Displays the priority level specified in the Action Events dialog box or Detailed Settings of Interrupt Events dialog box [Simulator] .

2. Click this line to display the detailed information of the lower lines.









(3) [Comment] area



The user can write comments for each event that has been set.

To input comments, click on this area, or select [Edit Comment] from the context menu after selecting the event in which you want to input comments, and then input directly the desired text from the keyboard (the edit mode is cancelled by pressing down the [Esc] key).

After editing the comments, complete the editing by pressing the [Enter] key or moving the focus to outside the edit region. Up to a maximum of 256 characters can be inputted for the comments, and this is saved as the settings of the user during use.

[Toolbar]

	Deletes the selected event. Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).
	Displays events related to the Hardware Break (default).
 [E1] [E20]	Displays events related to the Software Break (default).
	Displays events related to the trace (default).
 [E1] [E20]	Displays events related to the timer (default). [Simulator] This button is disabled.
	Displays events related to the action event (Printf event/Interrupt event) (default).
	Displays events related to the built-in event (Unconditional Trace event/Run-Break Timer event) (default).
	Opens the Editor panel and jumps to the source line corresponding to the address where the selected event ^{Note} is being set.

	Opens the Disassemble panel (Disassemble1) and jumps to the disassemble results corresponding to the address where the selected event ^{Note} is being set.
	Opens the Memory panel (Memory1) and jumps to the memory corresponding to the address where the selected event ^{Note} is being set.

Note This button is valid for events other than trace events, timer measurement events and built-in events (unconditional trace events/Run-Break Timer events).

[[Edit] menu (Events panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Events panel (other items are all invalid).

Delete	Deletes the selected event. Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).
Select All	Selects all the events displayed on the panel.
Find...	Opens the Find and Replace dialog box, with its [Find in Files] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Replace in Files] tab selected.

[Context menu]

Enable Event	Enables the selected event (valid state). Note that this item is invalid if the selected event is a valid state.
Disable Event	Disables the selected event (invalid state). Note that this item is invalid if the selected event is an invalid state.
Delete	Deletes the selected event. Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).
Select All	Selects all the events of this panel.
View Select	The following cascade menus are displayed to limit the event type to be displayed. All of the items have been selected by default.
Hardware Break	Displays events related to the Hardware Break.
Software Break	Displays events related to the Software Break.
Timer Event	Displays events related to the timer.
Trace Event	Displays events related to the trace.
Action Event	Displays events related to the action event (Printf event).
Built-in Event	Displays events related to the built-in event (Unconditional Trace event/Run-Break Timer event).
Timer Settings	The following cascade menus are displayed to do the settings related to the timer. However, this item is valid only when a timer-related event has been selected.
Init Timer	Initializes the timer used by the selected event (except for the Run-Break Timer).
Nanosecond	Displays the Timer Results of the selected event in nanosecond units.
Microsecond	Displays the Timer Results of the selected event in microsecond units.
Millisecond	Displays the Timer Results of the selected event in millisecond units.
Second	Displays the Timer Results of the selected event in second units.
Minute	Displays the Timer Results of the selected event in minute units.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the memory corresponding to the address where the selected event Note1 is being set.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1) and jumps to the disassemble results corresponding to the address where the selected event Note1 is being set.
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the address where the selected event Note1 is being set.
Edit Condition ...	Opens the detailed settings dialog box corresponding to a selected event Note2 . If a Trace event or combination break is selected, the Combination Condition dialog box [E1][E20] is opened. If a Printf event is selected, the Action Events dialog box is opened.
Edit Comment	Sets to the edit mode to input comments for the selected event. When comments are already present, all of that character string is set to a select state.

Notes 1. Events other than Trace events, Timer Result events and built-in events (Unconditional Trace events/Run-Break Timer events) can be objects of this button.

2. [Edit Condition ...] is valid for the following events.

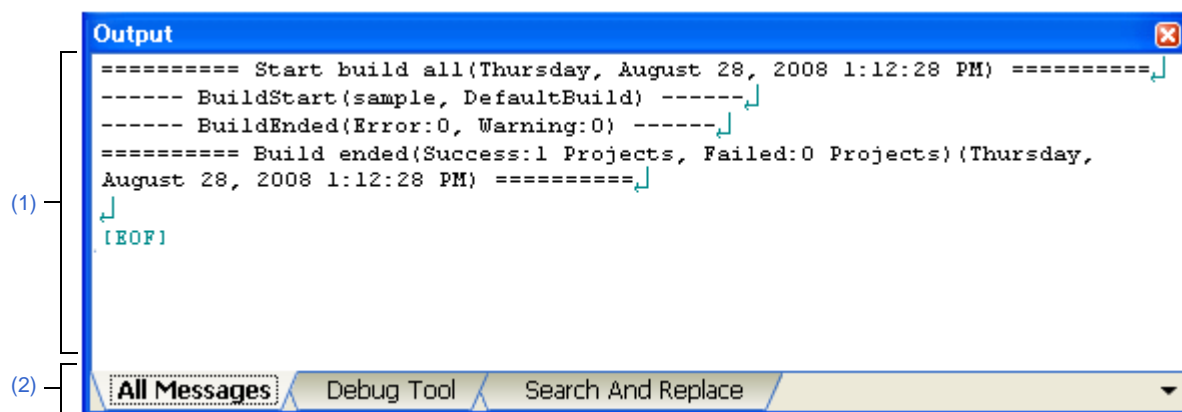
- Hardware breaks, execution-related events, access-related events, and timer measurement events **[E1] [E20]**

Output panel

This facility displays the messages output by various components supplied with CubeSuite+ (e.g., design, build and analysis tools, including the debug tool), as well as display the results of global searches conducted using the Find and Replace dialog box and the output results by Printf events (see "2.16.1 Insert printf").

Messages are displayed separately on respective tabs classified by tools from which they are output.

Figure A-37. Output Panel



Here, the following items are explained.

- [How to open]
- [Description of each area]
- [[File] menu (Output Panel-Only Items)]
- [[Edit] menu (Output Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Output] from the [View] menu.

[Description of each area]

(1) Message area

Displays the messages output by each tool, search results, and the output results by Printf events.

During display of search results (global search), this area displays a new message after clearing previous messages each time a search is performed (except for the [All Messages] tab).

Note that messages are displayed in different colors by type of output message, as shown below. (The colors in which text and backgrounds are displayed depend on how the [General - Font and Color] category of the Option dialog box are set.

Type of message	Display example (Default)			Description
Normal message	AaBbCc	Text color	Black	Displayed when notifying any information.
		Background color	White	
Warning message	AaBbCc	Text color	Blue	Displayed when notifying any warning for the operation performed.
		Background color	Standard color	

Type of message	Display example (Default)		Description
Error message	AaBbCc	Text color	Red
		Background color	Light gray
			Displayed when unable to execute for a fatal error or operational mistake.

This area provides the following facilities.

(a) Tag jump

Double-click an output message, or move the caret to a message and then hit the [Enter] key. The [Editor panel](#) is opened, displaying the relevant line number of the relevant file.

This facility permits you to jump to the relevant line in error of the source file from the error messages output, for example, at build time.

(b) Displaying help

While the caret is present at the line showing a warning or an error message, select [Help on Messages] on context menu or press the [F1] key. Help for a message on that line is displayed.

(c) Saving logs

Choose [Save Output-tab name As ...] from the [File] menu. The [Save As dialog box](#) is opened, allowing you to save the whole content displayed on the currently selected tab to a text file (*.txt). (Messages on unselected tabs are not saved.)

(2) Tab select area

Select a tab showing the source from which a message is output.

The debug tool uses the following tabs.

Tab name	Description
All Messages	Displays the messages output by all components supplied with CubeSuite+ (e.g., design, build and analysis tools, including the debug tool). (This does not apply to the messages associated with execution of a rapid build.)
Debug Tool	Displays only the messages output by the debug tool, out of those output by various components supplied with CubeSuite+ (e.g., design, build and analysis tools, including the debug tool).
Find and Replace	Displays the results of global searches conducted from the Find and Replace dialog box.

Caution Even when a new message is output on some unselected tab, the panel does not have its tabs automatically switched to show the new message. In this case, the relevant tab is marked with an asterisk (*) at the beginning of its name, indicating that a new message has been output.

[[File] menu (Output Panel-Only Items)]

The [File] menu used exclusively for the output panel is as follows. (The other items are shared.)

However, all of these items are disabled during program execution.

Save Output-tab name	Saves the contents displayed on the currently selected tab to a text file (*.txt) that has been saved previously. (See "(c) Saving logs") If this item is selected for the first time after startup, the same operation as you've selected [Save Output-tab name As...] is performed. However, this is disabled during build execution.
----------------------	---

Save Output-file name As...	Opens the Save As dialog box to save the contents displayed on the currently selected tab to a specified text file (*.txt). (See "(c) Saving logs ")
-----------------------------	--

[[Edit] menu (Output Panel-Only Items)]

The [Edit] menu used exclusively for the output panel is as follows. (All other items are disabled.)

Copy	Copies a selected character string to the clipboard.
Select All	Selects all of the messages displayed on the currently selected tab.
Find...	Opens the Find and Replace dialog box, with its [Quick Find] tab selected.
Replace...	Opens the Find and Replace dialog box, with its [Replace in Files] tab selected.

[Context menu]

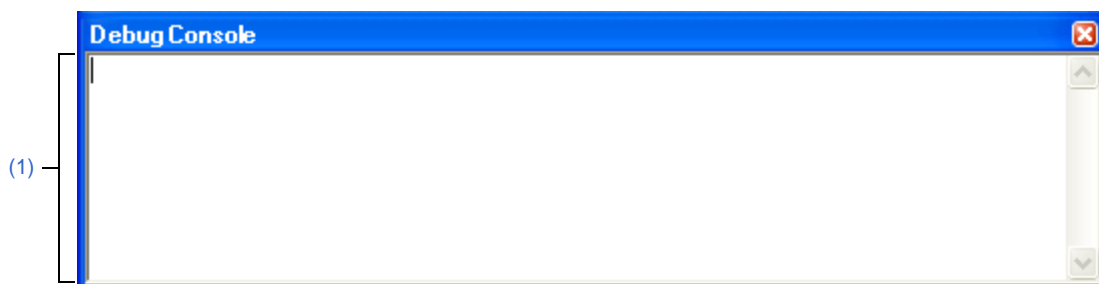
Copy	Copies a selected character string to the clipboard.
Select All	Selects all of the messages displayed on the currently selected tab.
Clear	Clears all of the messages displayed on the currently selected tab.
Tag Jump	Opens the Editor panel , with control jumping to the relevant line number in the file pertaining to the message at the caret position.
Stop Searching	Halts the search currently under execution. However, this menu is disabled when no searches are being executed.
Help for Message	Displays help for a message at the current caret position. However, this menu applies to only warning and error messages.

Debug Console panel

This panel exchanges data between the console and program. This is accomplished by executing a program which has standard library functions implemented in it.

Note that this panel can be opened only when CubeSuite+ is connected with the debug tool.

Figure A-38. Debug Console Panel



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[\[Edit\] menu \(Debug Console panel-only items\)\]](#)
- [\[Context menu\]](#)

[How to open]

- Select [\[Debug Console\]](#) from the [\[View\]](#) menu.

[Description of each area]**(1) Input/output area**

The standard library functions implemented in the program include, for example, the scanf function for reading the data entered from the keyboard and the printf function for outputting data.

Also, by specifying a COM port on the panel, it is possible to redirect standard input/output of the program to the specified COM port.

To use this function, the program must have the low-level interface routines provided by the debug tool implemented in it (see "[2.19 Using the Debug Console](#)").

Remark [Simulator]

For details about the input/output functions provided by the simulator, see "[APPENDIX B INPUT/OUTPUT FUNCTIONS](#)."

[[Edit] menu (Debug Console panel-only items)]

The [\[Edit\]](#) menu items provided exclusively on the Debug Console panel are as follows. (The other items are shared.)

Copy	Copies a selected character string to the clipboard.
Paste	Inserts the content of the clipboard into the caret position.
Select All	Selects all of the character strings displayed on this panel.

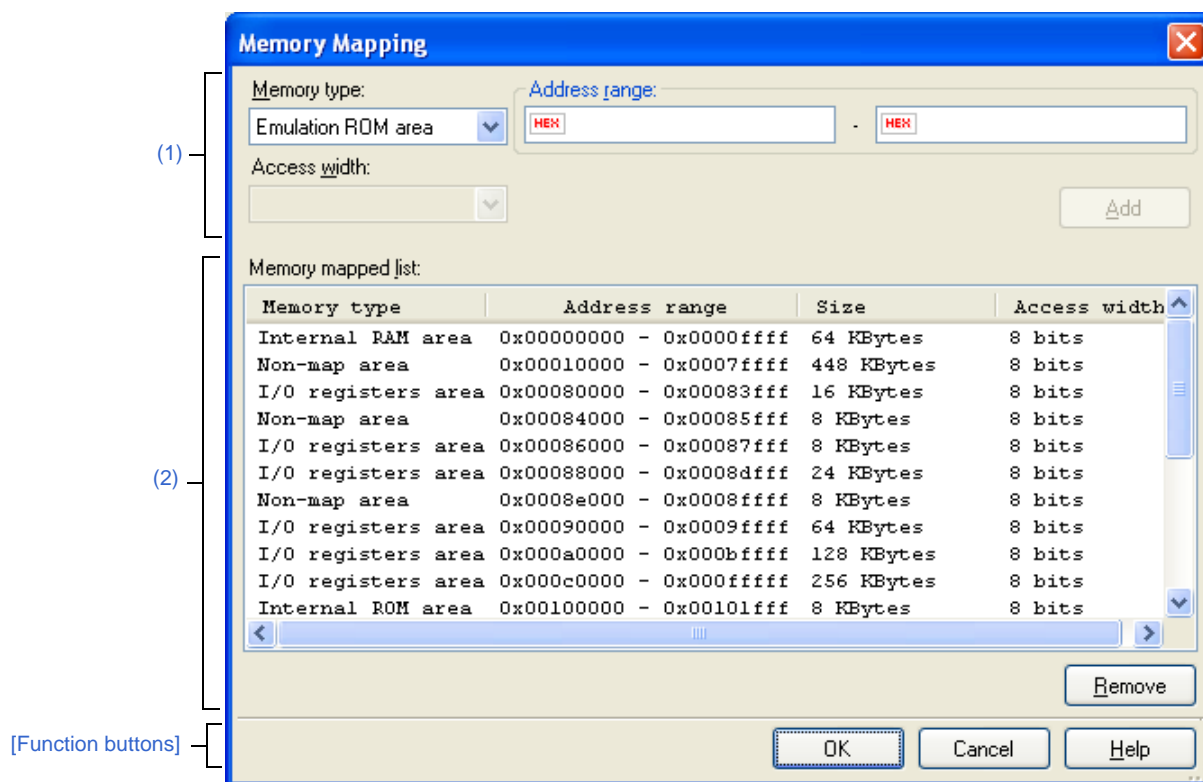
[Context menu]

Copy	Copies a selected character string to the clipboard.
Paste	Inserts the content of the clipboard into the caret position.
Clear	Clears the display of this panel.
Enable/Disable	Chooses to enable (default) or disable the debug console (a toggle switch). Note that when the debug console function is disabled, the panel's background color changes to gray.
COM Port...	Opens the Port Setting dialog box to set a COM port on the host machine to which communication from the microcontroller is redirected.
Log File...	Opens the Open Log File dialog box to save the displayed content of this panel to a specified log file (*.log). Logging begins.
Logging	Chooses to start (default) or stop logging (a toggle switch). However, this item is disabled when no log files are specified.
Echo Back	Chooses to enable (default) or disable local echoback (a toggle switch). If local echoback is disabled, the input data is not output to this panel.

Memory Mapping dialog box [Simulator]

This dialog box is used to set the memory mapping for each type of memory.

Figure A-39. Memory Mapping Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Debug Tool Settings] tab of the Property panel, click the [...] button displayed by selecting one of the values of the [Memory mappings] property in the [Memory] category.

Caution This dialog box cannot be opened during execution of a program.

[Description of each area]

(1) Added memory mapping specification area

Specify the information for a newly added memory mapping.

(a) [Memory type]

Select the memory type for the memory mapping to be added from the following drop-down list.

The item selected by default differs depending on the debug tool to use.

Emulation ROM area	Adds emulation ROM area. Uses simulator alternative ROM.
Emulation RAM area	Adds emulation RAM area. Uses simulator alternative RAM.

Note Memory mapping can be added in the unit of 16 bytes. If you set it outside the 16-byte boundary, compensation will be made to meet the 16-byte boundary area which includes the area set after [OK] is clicked.

(b) [Address range]

Specify the start address and end address for the memory mapping to be added.

Directly input a hexadecimal number into the text box for each.

Note that you cannot add memory mappings to the areas that overlap the following memory types. (A message will appear if you click [Add] button in these areas.)

- [Internal ROM area]
- [Internal RAM area]
- [IO register area]

(c) [Access width]

Access width cannot be specified.

(d) Button

Button	Function
Add	Adds the content specified in this area to memory mapping. The added memory mapping is displayed in the [Memory mapped list] area . The changes will not take effect until the [OK] button is clicked.

(2) [Memory mapped list] area

(a) List display

Information about the memory mapping added in the [Added memory mapping specification area](#) and the microcontroller's internal memory mapping is displayed. This area cannot be edited.

Memory type	Following memory types are displayed: <ul style="list-style-type: none"> - Internal ROM area - Internal RAM area - I/O register area - Emulation ROM area - Emulation RAM area - Non-map area
Address range	Displays the address range as <Start address> - <End address>. Display is fixed as "0x"-prefixed hexadecimal numbers.
Size	Displays size as a decimal number (unit: bytes/Kbytes).
Access width	Displays the access width (unit: bits) ^{Note} .

Note Since access width is not supported in the simulator, a fixed value (8 bit) will be displayed. The simulation execution time will not be affected by the access width value.

(b) Button

Button	Function
Remove	Deletes the memory mapping selected in this area. The memory areas that can be deleted are the Emulation ROM area or the Emulation RAM area (the microcontroller's internal memory mapping cannot be deleted).

[Function buttons]

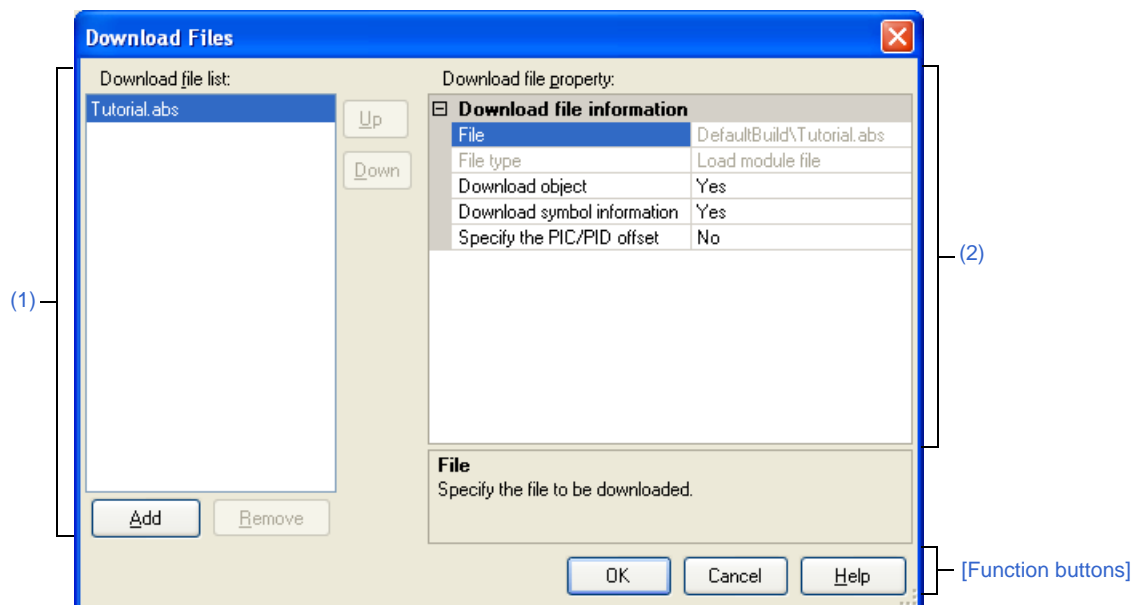
Button	Function
OK	Sets the currently specified memory mapping to the debug tool and closes this dialog box.
Cancel	Cancels memory mapping changes and closes this dialog box.
Help	Displays the help for this dialog box.

Download Files dialog box

This dialog box selects a file to download and sets download conditions (see Section "2.5 Download and Upload").

The files specified in the project (main project or sub-project) as the subject to build are automatically registered as the subject files to be downloaded (not removable).

Figure A-40. Download Files Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Select the [Download files] property in the [Download] category on the [Download File Settings] tab of the Property panel and then click the [...] button that is displayed.

Caution This dialog box cannot be opened during program execution.

[Description of each area]

(1) [Download File List] area

(a) Displaying a list

This area displays a list of file names to download. By default, the file names specified in the project (main project or sub-project) as the subject to build are displayed (not removable).

Files are downloaded in the order in which they are listed here.

To add a new download file, click the [Add] button in this area and then specify the download conditions for the file to be added in the [Download file property] area.

(b) Buttons

Button	Function
Up	Moves a selected file one line up. However, this button is disabled when the file at the top of the list or a file specified as the subject to build in the project is selected.
Down	Moves a selected file one line down. However, this button is disabled when the file at the bottom of the list or a file specified as the subject to build in the project is selected.
Add	Adds one blank item ("-") to the list, with the item selected. In the [Download file property] area , specify the download conditions for the file to be added. However, this button is disabled when 20 or more files are already registered.
Remove	Removes a selected file from the list. However, the files specified as the subject to build in the project cannot be removed.

- Remarks 1.** Place the mouse cursor at a file name, and the path information for the subject file is displayed in a popup box.
- 2.** The order in which files are listed can be changed by dragging any file name up or down in the list with the mouse. However, the files specified as the subject to build in the project cannot have their order in the list changed.

(2) [Download file property] area

(a) [Download file information]

This section displays download conditions or changes of settings made for a file selected in the [\[Download File List\] area](#).

Also, if a new download file is added using the [Add] button, this section may be used to specify download conditions for the file added.

File	Specify a file to download	
	Default	File name (However, blank when newly added)
	How to change	Enter directly from the keyboard, or specify in the Select Download File dialog box that is opened by clicking the [...] button ^{Note 1} that is displayed at the right edge of the column when this item is selected.
	Specifiable value	See " Table 2-2. Downloadable File Formats ". Specifiable in up to 259 characters
File type	Specify the file format of a file to download.	
	Default	Load module file
	How to change	Select from the drop-down list.
	Specifiable value	One of the following: - Load module file - Hex file - S record file - Binary data file

Offset	This item is displayed only when the files to download are in hex format. Specify an offset value from the address at which download of a specified file begins.	
	Default	0
	How to change	Enter directly from the keyboard.
	Specifiable value	Hexadecimal values from 0x0 to 0xFFFFFFFF
Start address	This item is displayed only when the files to download are in binary data format. Specify the start address from which a specified file is downloaded.	
	Default	0
	How to change	Enter directly from the keyboard.
	Specifiable value	Hexadecimal values from 0x0 to 0xFFFFFFFF
Download object	This item is displayed only when the files to download are in load module format. Specify whether or not to download object information from a specified file.	
	Default	Yes
	How to change	Select from the drop-down list.
	Specifiable value	Yes Object information is downloaded.
		No Object information is not downloaded.
Download symbol information	This item is displayed only when the files to download are in load module format. Specify whether or not to download symbol information from a specified file ^{Note 2} .	
	Default	Yes
	How to change	Select from the drop-down list.
	Specifiable value	Yes Downloads symbol information.
		No Does not download symbol information.
Specify the PIC/PID offset	Specify whether to change the positions of PIC (Position Independent Code) and PID (Position Independent Data) areas of the load modules to download from those specified during the creation of load modules. When "Yes" is selected, "PIC offset" and "PID offset" will appear as sub-items.	
	Default	No
	How to change	Select from the drop-down list.
	Specifiable value	Yes PIC/PID offset is specified ^{Note 3} .
		No PIC/PID offset is not specified.
PIC Offset	Input the offset values from the address specified at the time of load module creation. For instance, if you enter "1000" here when the start address of the program section is 0x1000, the corresponding section will be downloaded to 0x2000.	
	Default	0
	How to change	Enter directly from the keyboard.
	Specifiable value	Hex number between 0x0 and 0xFFFFFFFF

PID Offset	Input the offset values to the PID register specified at the time of load module creation. For instance, if you want to set 0x200 to the PID register when executing the load module, enter "200" here.	
	Default	0
	How to change	Enter directly from the keyboard.
	Specifiable value	Hex number between 0x0 and 0xFFFFFFFF

- Notes**
1. If any file as the subject to build in the project is selected in the [\[Download File List\] area](#), or while the program is under execution, the [...] button is not displayed.
 2. Unless symbol information is downloaded, source-level debugs cannot be performed.
 3. Proper debug operation is not guaranteed when you have selected "Yes" for load modules that were created without using PIC/PID function (see Section "2.7 Usage of PIC/PID Function").

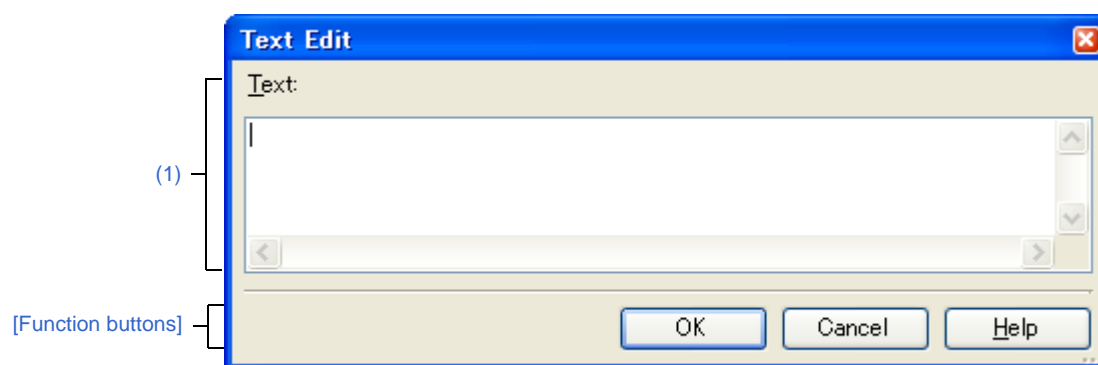
[Function buttons]

Button	Function
OK	Terminates download file settings and closes this dialog box.
Cancel	Nullifies download file changes made and closes this dialog box.
Help	Displays help for this dialog box.

Text Edit dialog box

This dialog box is used to input/modify character strings.

Figure A-41. Text Edit Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [\[Hook Transaction Settings\]](#) tab of the [Property panel](#), click the [...] button displayed by selecting one of the property in the [\[Hook Transaction Settings\]](#) category.

[Description of each area]**(1) [Text] area**

Input/modify character strings in this area.

[Function buttons]

Button	Function
OK	Sets the input character strings to the caller panel/dialog box and closes this dialog box.
Cancel	Closes this dialog box.
Help	Displays help for this dialog box.

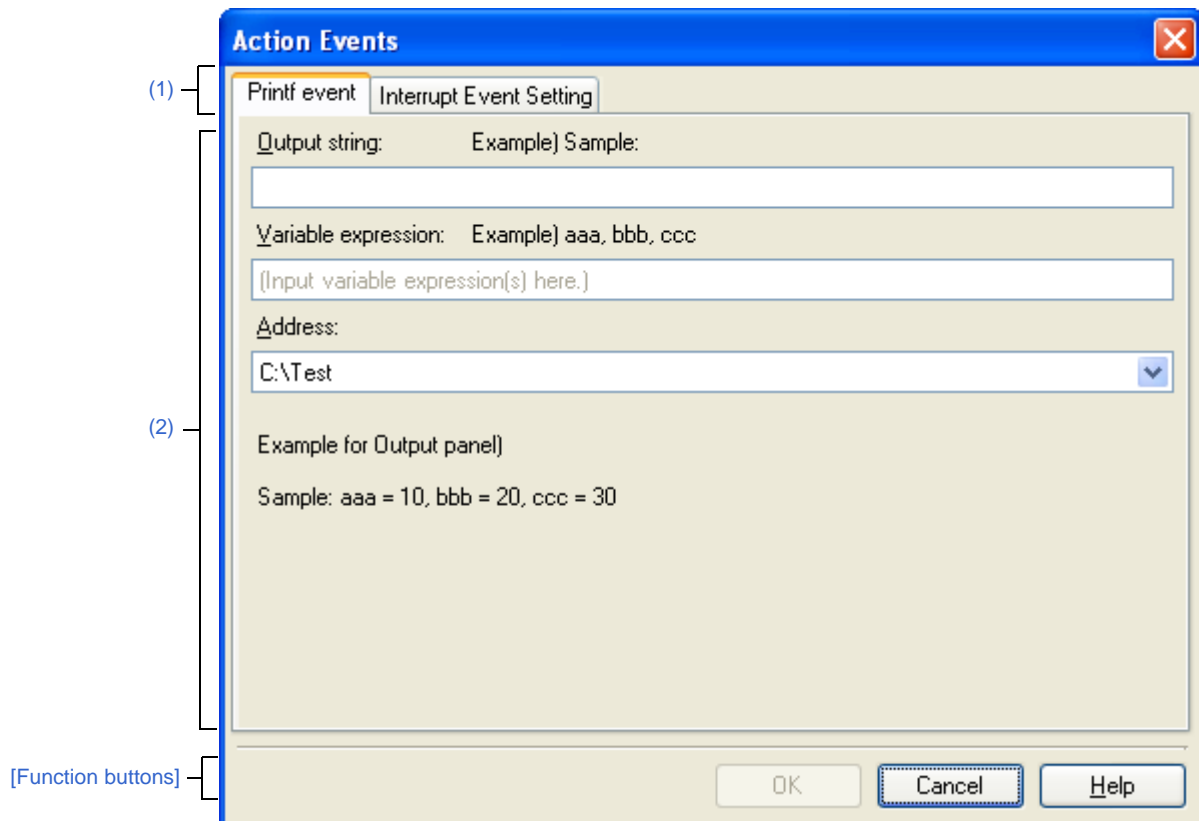
Action Events dialog box

You can set action events in this dialog box (see "2.16 Set an Action into Programs").

This dialog box can be opened only when you are connected to the debug tool.

Caution Also see "2.17.7 Points to note regarding event setting" for details on action event settings (e.g. limits on the number of enabled events).

Figure A-42. Action Events Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Editor panel](#), move the caret to the line where you wish to set an action event, then select [Register Action Event...] from the context menu.
- On the [Disassemble panel](#), move the caret to the address where you wish to set a Printf event, then select [Register Action Event...] from the context menu.
- On the [Events panel](#), select [Edit the conditions...] from the context menu after selecting the action event.

[Description of each area]

(1) [Tab selection] area

You can switch action events to be registered by the selection of a tab. This dialog box has the following tabs.

- [\[Printf event\] tab](#)
- [\[Interrupt event setting\] tab](#) [\[Simulator\]](#)

Caution When you open this dialog box by selecting **[Edit the conditions...]** from the context menu, this area will be hidden.

(2) [Event conditions settings] area

You can set detailed conditions for each action event in this area. See the section of the corresponding tab for the details of the setting procedure.

[Function buttons]

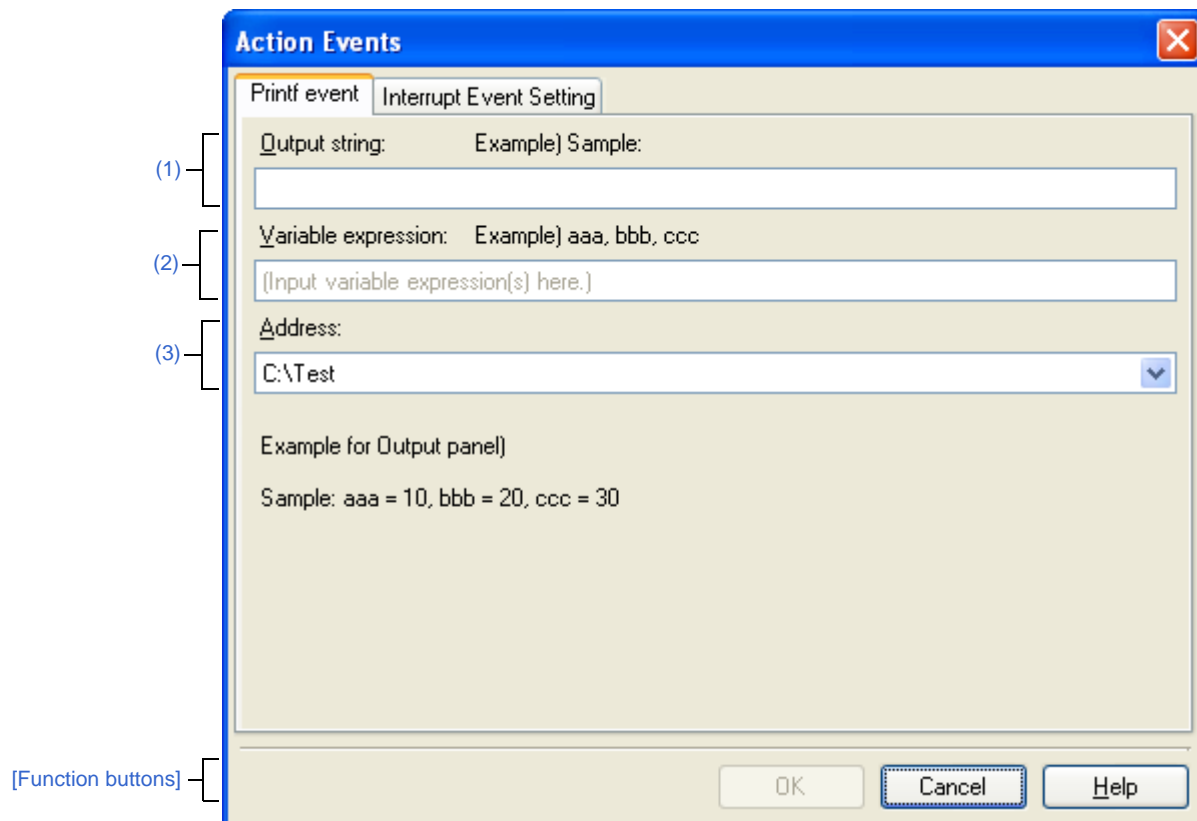
Button	Function
OK	Ends the setting of the action event and sets the specified action event at a specified location.
Cancel	Cancels the action event setup and closes this dialog box.
Help	Displays the help for this dialog box.

[Printf event] tab

This tab is used to configure Printf events as action events (see "2.16.1 Insert printf").

A Printf event momentarily stops the execution of the program at a specified location, and executes the printf command via software processing. When a Printf event is set, the program momentarily stops immediately before executing the command at the location where this event is set, and the value of the variable expression specified in this dialog box is output to the [Output panel](#).

Figure A-43. Action Events Dialog Box: [Printf Event] Tab



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Editor panel](#), move the caret to the line where you wish to set a Printf event, then select [Register Action Event...] from the context menu.
- On the [Disassemble panel](#), move the caret to the address where you wish to set a Printf event, then select [Register Action Event...] from the context menu.
- On the [Events panel](#), select [Edit the conditions...] from the context menu after selecting the Printf event.

[Description of each area]**(1) [Output string] area**

Type in the string to add to the [Output panel](#) directly via the keyboard (up to 1024 characters).

Note that the output string can only be one line (spaces allowed).

(2) [Variable expression] area

Specify the variable expression(s) for the Printf event.

Type a variable expression directly into the text box (up to 1024 characters).

You can specify up to 10 variable expressions for a single Printf event by separating them with commas (",").

If this dialog box is opened with a variable expression selected in the [Editor panel](#) / [Disassemble panel](#), the selected variable expression appears as the default.

The basic input format that can be specified as variable expressions and the values output by Printf event are as follows:

Table A-14. Relationship between Variable Expressions and Output Value (Printf Event)

Variable Expression	Output Value
Name of a C/C++ language variable ^{Note 1}	Value of a C/C++ language variable
<i>Variable expression</i> [<i>Variable expression</i>]	Element of an array
<i>Variable expression</i> .Member name ^{Note 2}	Value of a structure/union/class member
<i>Variable expression</i> -> Member name ^{Note 2}	Value of a structure/union/class member that pointer to
Variable expression.*Cast expression	Value of the pointer to a member variable
Variable expression->*Cast expression	Value of the pointer to a member variable
* <i>Variable expression</i>	Value of a pointer variable
& <i>Variable expression</i>	Location address
(Type name) <i>Variable expression</i>	Value cast into a specified type
Name of the CPU register	Value of the CPU register
IOR name	IOR value
Label name ^{Note 3} , EQU symbol name ^{Note 3} , and [immediate value]	Values of a label, an EQU symbol, and immediate address

Notes 1. C89, C99, or C++ language variable

2. When using a member variable of a base class, specify the scope before the member name (e.g. variable.BaseClass::member).

3. If the label name or EQU symbol name includes a "\$," be sure to enclose the name in "{ }" (Example: {\$Label}).

Any imaginary number must be multiplied by an uppercase "I" (e.g. 1.0 + 2.0*I). When you specify the CPU register name "I", add ":REG" (e.g. I:REG) to distinguish it from the keyword "I".

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see ["2.20.2 Symbol name completion function"](#)).

(3) [Address] area

Specify the address at which to set the Printf event.

You can either type address expressions directly into the text boxes (up to 1024 characters), or select them from the input history via the drop-down list (up to 10 items). By default, address of the presently specified location is displayed.

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 [Symbol name completion function](#)").

Note that the output result format by the Printf event in the [Output panel](#) are as follows:

Figure A-44. Output Result Format of Printf Event

```
Specified characters Variable expression 1 = Value 1, Variable expression 2 = Value
2, Variable expression 3 = Value 3, ...
```

<i>Specified characters</i>	Characters specified with [Output string]
<i>Variable expression 1 - 10</i>	Characters specified with [Variable expression]
<i>Value 1 - 10</i>	Value of variable corresponds to " <i>Variable expression 1 - 10</i> " The value is displayed in a format that matches the variable type (see Table A-10. Display Format of Watch Expressions (Default)). ("?")will be displayed when the specified variable expression cannot be acquired. The value is also displayed in hexadecimal in bracket "()". (If a hexadecimal value cannot be displayed, "-" will be shown instead.)

[Function buttons]

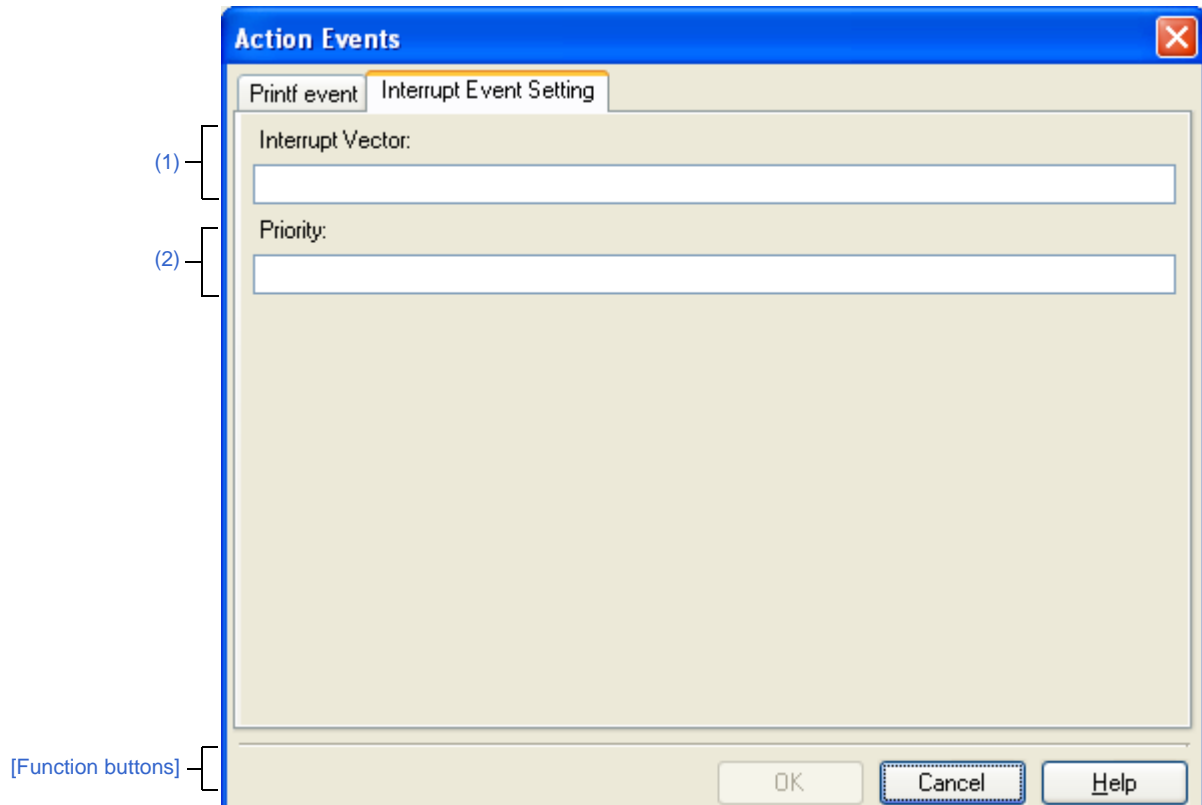
Button	Function
OK	Finishes configuring the Printf event, and sets it at the caret position in the Editor panel/Disassemble panel .
Cancel	Cancels the Printf event setup and closes this dialog box.
Help	Displays the help for this dialog box.

[Interrupt event setting] tab [Simulator]

You can set an interrupt event in this tab (see "2.16.2 Insert an interrupt event [Simulator]").

An interrupt event is a function with which you can generate an interrupt request at a specified point. When an interrupt event is set, an interrupt request will occur immediately before the execution of an instruction for which the event is set. Once the interrupt request is accepted by the CPU, interrupt exception will take place.

Figure A-45. Action Events Dialog Box: [Interrupt event setting] Tab



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Editor panel](#), move the caret to the line where you wish to set an interrupt event, then select [Register Action Event...] from the context menu.
- On the [Disassemble panel](#), move the caret to the address where you wish to set an interrupt event, then select [Register Action Event...] from the context menu.
- On the [Events panel](#), select [Edit the conditions...] from the context menu after selecting the interrupt event.

[Description of each area]

(1) [Interrupt Vector] area

Specify the interrupt vector by directly entering a corresponding number between 0 and 255.

(2) [Priority] area**- [RX610 Group]**

Specify the priority order by directly entering a number between 1 and 8.

When 8 is specified, the event is handled as a fast interrupt.

- [Non-RX610 Group]

Specify the priority order by directly entering a number between 1 and 16.

When 16 is specified, the event is handled as a fast interrupt.

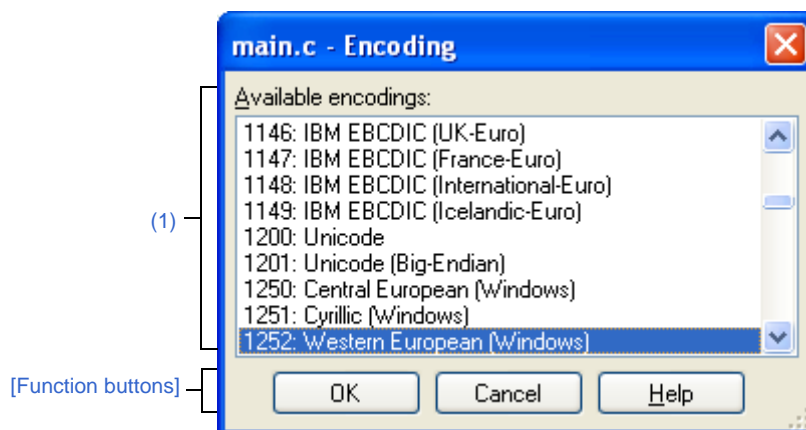
[Function buttons]

Button	Function
OK	Finishes configuring the interrupt event, and sets it to the line/address at the caret position in the Editor panel/Disassemble panel .
Cancel	Cancels the interrupt event setup and closes this dialog box.
Help	Displays the help for this dialog box.

Encoding dialog box

This dialog box is used to select a file-encoding.

Figure A-46. Encoding Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- From the [File] menu, open the [Open File dialog box](#) by selecting [Open with Encoding...], and then click the [Open] button in the dialog box.

[Description of each area]

(1) [Available encodings]

Select the encoding to be set from the drop-down list.

All code-pages and encodings supported by the OS are displayed in alphabetical order.

Note that the same encoding and encoding which are not supported by the current OS will not be displayed.

The default file-encoding in the [General - Text Editor] category of the [Option dialog box](#) is selected by default.

[Function buttons]

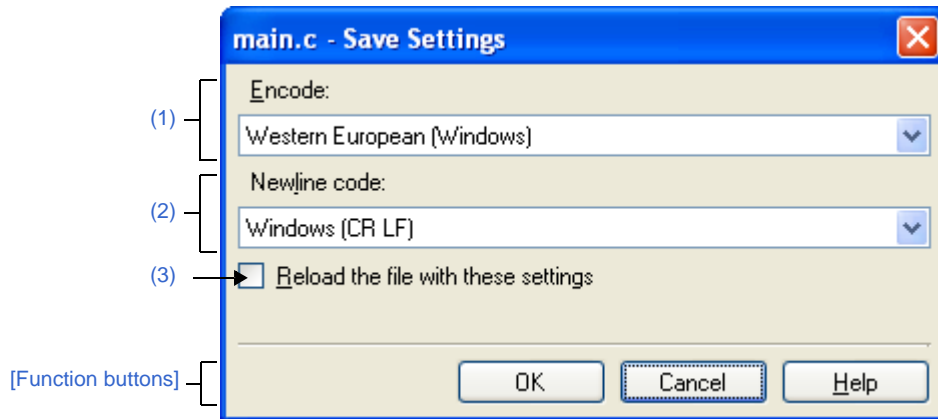
Button	Function
OK	Opens the selected file in the Open File dialog box using a selected file encoding.
Cancel	Not open the selected file in the Open File dialog box and closes this dialog box.
Help	Displays help for this dialog box.

Save Settings dialog box

This dialog box is used to set the encoding and newline code of the file that is being edited on the [Editor panel](#).

Remark The target file name is displayed on the title bar.

Figure A-47. Save Settings Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- Focus the [Editor panel](#), and then select [*File name* Save Settings...] from the [File] menu.

[Description of each area]

(1) [Encode] area

Select the encoding to be set from the drop-down list.

The items of the drop-down list are displayed according to the following sequence.

Note that the same encoding and encoding which are not supported by the current OS will not be displayed.

- *Current encoding of the file (default)*
- *Default encoding of the current OS*
- *Most recently used encodings (maximum 4)*
- *Popular encodings for current locale*
(e.g. for United States locale it will be:
 - Western European (Windows)
 - Unicode (UTF-8)
- *All other encodings supported by the OS (in alphabetical order)*

(2) [Newline code] area

Select the newline code to be set from the drop-down list.

You can select any of items below.

- Windows (CR LF)
- Macintosh (CR)
- Unix (LF)

An active newline entry is selected by default.

(3) [Reload the file with these settings]

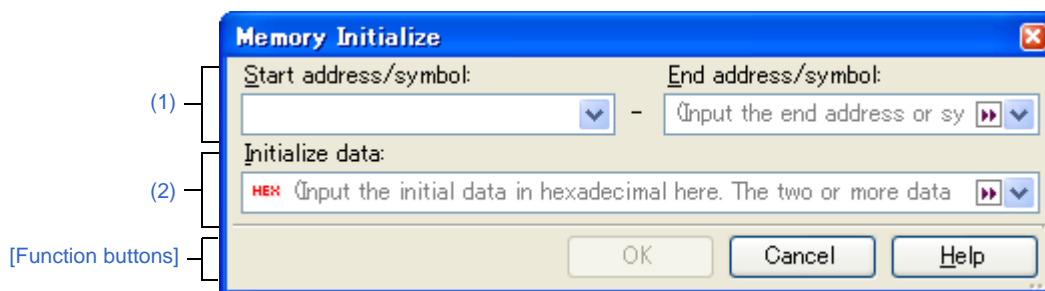
<input checked="" type="checkbox"/>	Reloads the file with the selected encoding and newline code when the [OK] button is clicked.
<input type="checkbox"/>	Does not reload the file when the [OK] button is clicked (default).

[Function buttons]

Button	Function
OK	Sets the selected encoding and newline code to the target file and closes this dialog box. If [Reload the file with these settings] is selected, sets the selected encoding and newline code to the target file and reloads the file. And then closes this dialog box.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help for this dialog box.

Memory Initialize dialog box

This dialog box initializes memory values. (See "(6) Collectively changing (initializing) memory contents".)
A pattern of specified initialization data is written into a specified address range of memory repeatedly.

Figure A-48. Memory Initialize Dialog Box

Here, the following items are explained.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Choose [Initialize ...] from the context menu on the [Memory panel](#).

[Description of each area]**(1) Range specification area**

Specify an address range of memory whose values need to be initialized in [Start address/symbol] and [End address/symbol]. Enter address expressions directly in the respective text boxes (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries).
The calculation results of the input address expressions are handled respectively as the start address and end address.
No address values can be specified that are greater than the microcontroller's address space.

Caution Note that an address range that covers areas with different endians cannot be specified.

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 [Symbol name completion function](#)").

(2) [Initialize data] area

Specify the initialization data to be written into memory.
To specify initialization data, enter a hexadecimal value directly in the text box or select an input history item from the drop-down list (up to 10 history entries).
To specify multiple pieces of initialization data, specify a maximum of 16 pieces of up-to-4-byte data (8 characters) by separating each with a space.
Each piece of initialization data are interpreted as comprising 1 byte in units of 2 characters from the tail end of the character string. If the data consists of an odd number of characters, the first character in it is assumed to be comprising 1 byte.

Note that if the data consists of 2 bytes or more, it is converted, before being written into the target memory, to an appropriate byte sequence that suits the project's endian as shown below.

Input character string (initialization data)	Written-in image (in bytes)	
	Little endian	Big endian
1	01	01
0 12	00 12	00 12
00 012 345	00 12 00 45 03	00 00 12 03 45
000 12 000345	00 00 12 45 03 00	00 00 12 00 03 45

[Function buttons]

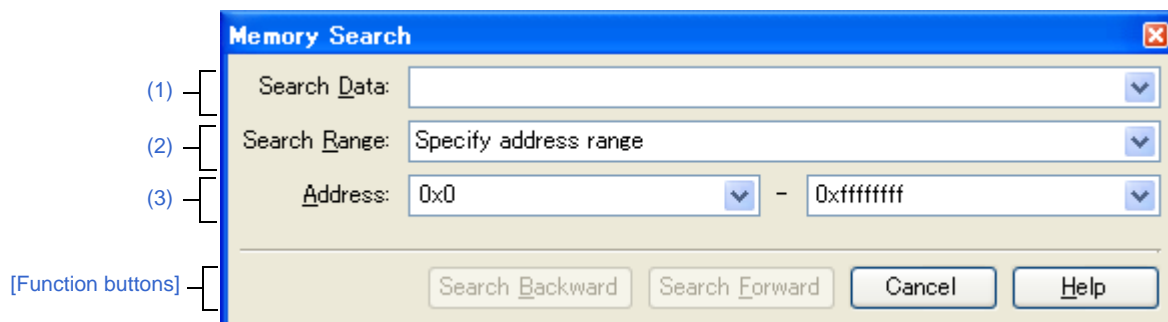
Button	Function
OK	Writes a pattern of specified initialization data into a specified address range of memory repeatedly. (If the end address is reached in the middle of this pattern, the write process is terminated.)
Cancel	Nullifies settings for memory value initialization and closes this dialog box.
Help	Displays help for this dialog box.

Memory Search dialog box

This dialog box searches for memory values. (See "(5) Searching for memory contents")

A search is performed in either the [Memory value area](#) or [Character string area](#) in which the caret on the [Memory panel](#) was present immediately before this dialog box is opened.

Figure A-49. Memory Search Dialog Box



Here, the following items are explained.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- Choose [\[Find...\]](#) from the context menu on [Memory panel](#).

[Description of each area]

(1) [Search Data] area

Specify the data to search.

Enter directly in the text box (specifiable in up to 256 bytes) or select an input history item from the drop-down list (up to 10 history entries).

If the subject of search is in the [Memory value area](#) of the [Memory panel](#), the data needs to be entered in the same form (numeral system and size) as displayed in that area.

Also, the subject of search is in the [Character string area](#), it is necessary to specify a character string as the data to search. The specified character string, before being searched, is converted to data in appropriate encode form in which data are displayed in that area.

Note that if any memory value was selected immediately before this dialog box was opened, then the selected value is displayed by default.

(2) [Search Range] area

Select a range in which to search from the drop-down list below.

Specify address range	A search is conducted within the address range specified by [Address] area .
<i>Memory mapping</i>	<p>A search is conducted within the selected range of memory mapping.</p> <p>This list item displays memory mappings individually (except non-mapped areas) that are displayed in the Memory Mapping dialog box [Simulator].</p> <p>Display form: <Memory type> <Address range> <Size></p>

(3) [Address] area

This item is valid only when [Specify Address Range] is selected in the [\[Search Range\] area](#).

Specify the "start address" and "end address" to set the address range in which a memory value is searched.

Directly enter address expressions in the respective text boxes (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries).

The calculation results of the entered address expressions are handled respectively as the start address and end address.

However, searchable addresses are limited to the upper-limit address of the program space (0xFFFFFFFF).

Also, no address values can be specified that are greater than the value representable by 32 bits.

- Remarks 1.** By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 [Symbol name completion function](#)").
2. If the "start address" text box is blank, address "0x0" is assumed.
 3. If the "end address" text box is blank, the upper-limit address of the microcontroller's address space is assumed.

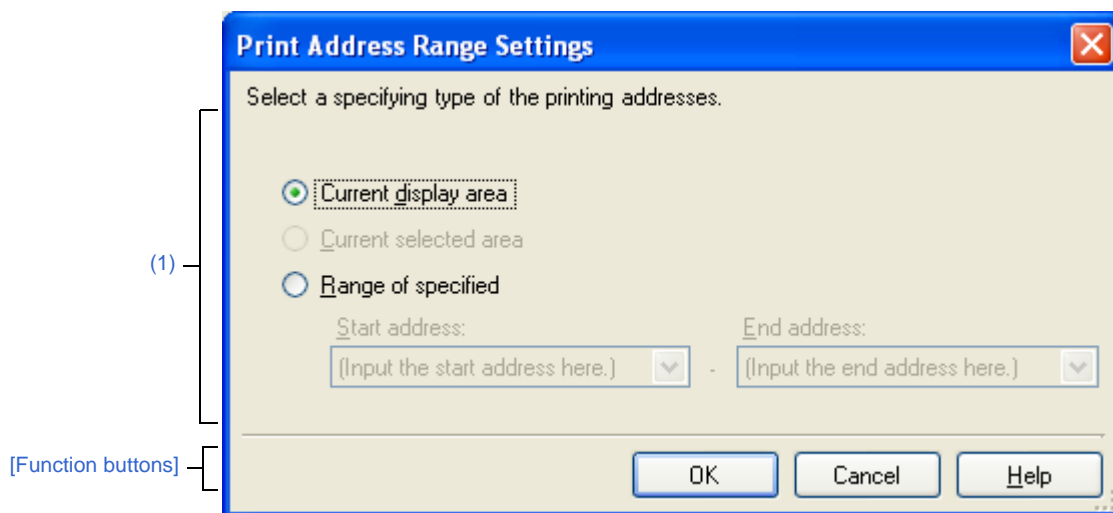
[Function buttons]

Button	Function
Search Backward	<p>Performs a search in the direction toward smaller addresses within the range specified in the [Search Range] area and [Address] area. The searched spot is placed in selected state on Memory panel.</p> <p>However, if an invalid value is specified, or when the program is under execution, a message is displayed and a search for memory value is not performed.</p> <p>Also, if the Memory panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.</p>
Search Forward	<p>Performs a search in the direction toward larger addresses within the range specified in the [Search Range] area and [Address] area. The searched spot is placed in selected state on Memory panel.</p> <p>However, if an invalid value is specified, or when the program is under execution, a message is displayed and a search for memory value is not performed.</p> <p>Also, if the Memory panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.</p>
Cancel	Nullifies settings for a search of memory value and closes this dialog box.
Help	Displays help for this dialog box.

Print Address Range Settings dialog box

This dialog box is used to specify the address range to print the contents of the [Disassemble panel](#).

Figure A-50. Print Address Range Settings Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Disassemble panel](#), select [Print...] from the [File] menu.

[Description of each area]

(1) Range specification area

Select a range to print from the following option buttons.

(a) [Current display area] (default)

Prints only the contents of the [Disassemble panel](#) currently being displayed.

(b) [Current selected area]

Prints only the range currently being selected in the [Disassemble panel](#).

Note, however, that this option button will be invalid when nothing is selected in the Disassemble panel.

(c) [Range of specified]

Specify the range of address to print via [Start address] and [End address]. You can either type address expressions directly into the text boxes (up to 1024 characters), or select them from the input history via the drop-down list (up to 10 items).

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").

[Function buttons]

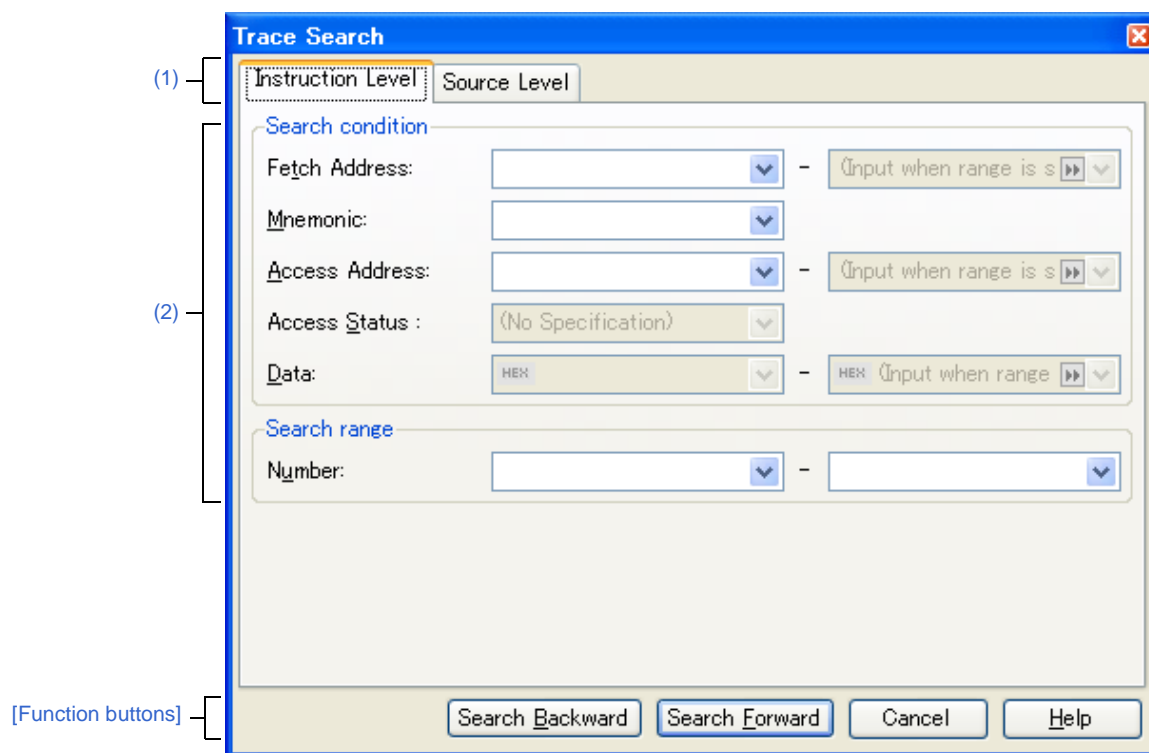
Button	Function
OK	Closes this dialog box and opens the Windows dialog box to print the contents of the specified range of Disassemble panel .
Cancel	Cancels the range specification and closes this dialog box.
Help	Displays help for this dialog box.

Trace Search dialog box

This dialog box searches for trace data (see Section "2.13.7 Searching for trace data").

Before performing a search, it is possible to choose whether the search is made at the instruction level or source level.

Figure A-51. Trace Search Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Select the toolbar button  on the Trace panel.
- Choose [Find...] from the context menu on the Trace panel.

[Description of each area]

(1) Tab selection area

Selecting a tab switches the level at which a search is performed.

This dialog box has the following tabs:

- [Instruction Level] tab
- [Source Level] tab

(2) Search condition setting area

Set specific conditions under which a search is performed.

For details about display contents and on how to set, see the section in which the relevant tab is described.

[Function buttons]

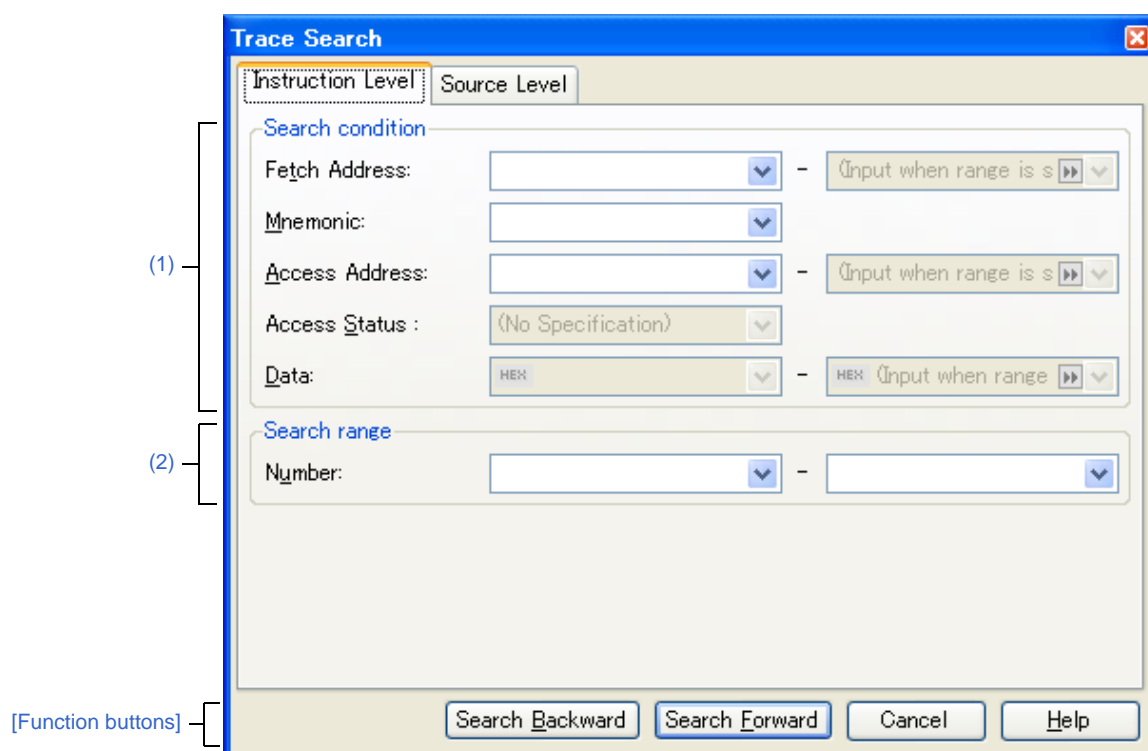
Button	Function
Search Backward	<p>Performs a search in the direction toward smaller addresses within a specified range.</p> <p>The searched spot is put in selected state on the Trace panel.</p> <p>However, if an invalid value is specified or when the program is under execution, a message is displayed and a search of trace data is not performed.</p> <p>Also, if the Trace panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.</p>
Search Forward	<p>Performs a search in the direction toward larger addresses within a specified range.</p> <p>The searched spot is put in selected state on the Trace panel.</p> <p>However, if an invalid value is specified or when the program is under execution, a message is displayed and a search of trace data is not performed.</p> <p>Also, if the trace panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.</p>
Cancel	<p>Nullifies the setting for a search of trace data and closes this dialog box.</p>
Help	<p>Displays help for this dialog box.</p>

[Instruction Level] tab

The acquired trace data is searched at the instruction level.

Caution If, while the **Trace panel** is displayed in the **Source display mode**, an instruction-level search is performed on this tab, the subject data cannot be searched correctly.
To perform an instruction-level search, make sure the Trace panel is displayed in the **Mixed display mode** or **Disassemble display mode**.


Figure A-52. Trace Search Dialog Box: [Instruction Level] Tab



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Select the toolbar button  on the **Trace panel**.
- Choose [Find...] from the context menu on the **Trace panel**.

[Description of each area]**(1) [Search condition] area****(a) [Fetch Address]**

Specify a fetch address, if needed as the search condition.

Enter an address expression directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The fetch address can also be specified as a range of addresses. In this case, enter address expressions in both the right and left text boxes to specify a range.

If the right-side text box is blank or labeled [(Input value when range is specified)], the fixed address specified in the left-side text box is used to perform a search.

Note that if any address value greater than the microcontroller's address space is specified, the high-order address value is masked when a search is performed.

Also, no address values can be specified that are greater than the value representable by 32 bits.

(b) [Mnemonic]

Specify a character string of instruction, if needed as the search condition.

The character string specified here is searched from within the [\[Source/Disassemble\]](#) area of the [Trace panel](#).

Enter an instruction directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-insensitive, and partial matches are also allowed.

(c) [Access Address]

Specify an access address, if needed as the search condition.

Enter an address expression directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The access address can also be specified as a range of addresses. In this case, enter address expressions in both the right and left text boxes to specify a range.

If the right-side text box is blank or labeled [(Input value when range is specified)], the fixed address specified in the left-side text box is used to perform a search.

Note that if any address value greater than the microcontroller's address space is specified, the high-order address value is masked when a search is performed.

Also, no address values can be specified that are greater than the value representable by 32 bits.

(d) [Access Status]

This item is enabled only when [\[Access Address\]](#) is specified.

Select the type of access from the drop-down list below.

When not limiting the type of access, select [No Specification].

(No Specification)
Read/Write
Read
Write

(e) [Data]

This item is enabled only when [\[Access Address\]](#) is specified.

Specify an accessed numeric value.

Enter a hexadecimal value directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The numeric value can be specified as a range of values. In this case, enter data in both the right and left text boxes to specify a range.

If the right-side text box is blank or labeled [(Input value when range is specified)], the fixed numeric value specified in the left-side text box is used to perform a search.

(2) [Search range] area**(a) [Number]**

Specify a range of trace data to search by numbers displayed in the [\[Number\] area](#) of the [Trace panel](#).

Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "*last number*" are specified.)

Enter a number in decimal notation directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

If the left-side text box is blank, the number "0" is assumed.

If the right-side text box is blank, the "*last number*" is assumed.

[Function buttons]

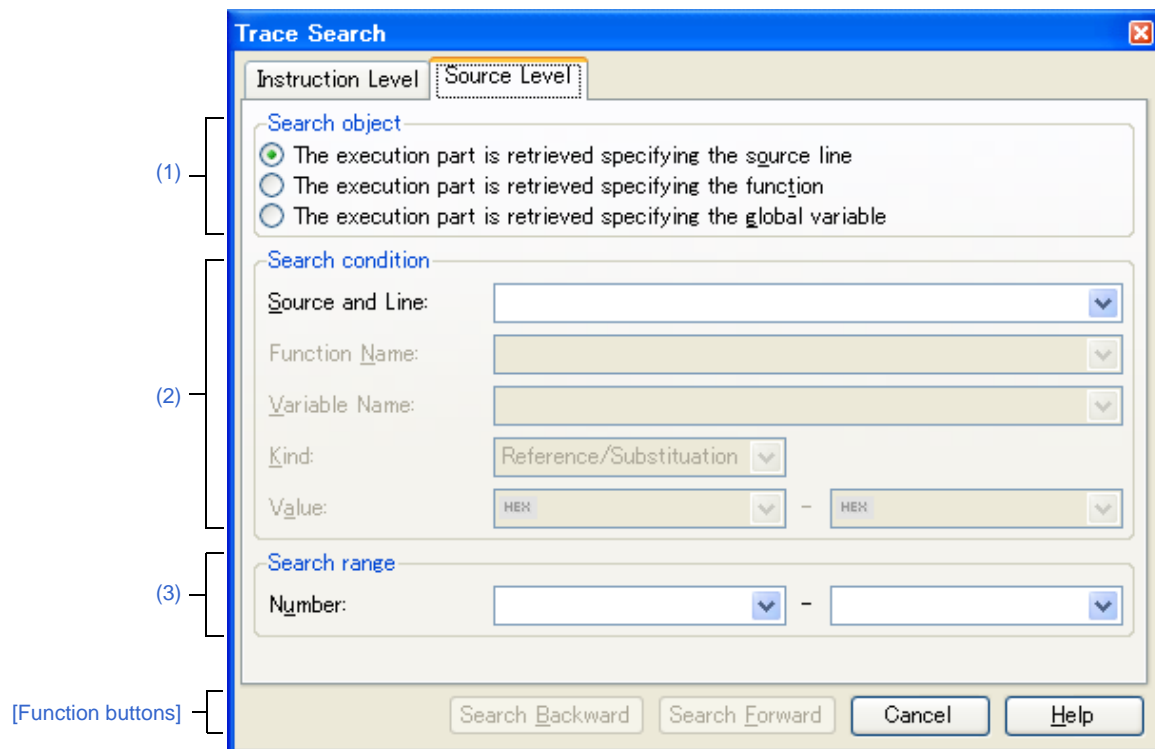
Button	Function
Search Backward	Performs a search in the direction toward smaller addresses within a specified range. The searched spot is put in selected state on the Trace panel . However, if an invalid value is specified, a message is displayed and a search of trace data is not performed. Also, if the trace panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.
Search Forward	Performs a search in the direction toward larger addresses within a specified range. The searched spot is put in selected state on the Trace panel . However, if an invalid value is specified, a message is displayed and a search of trace data is not performed. Also, if the trace panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.
Cancel	Nullifies the setting for a search of trace data and closes this dialog box.
Help	Displays help for this dialog box.

[Source Level] tab

The acquired trace data is searched at the source level.

Caution If, while the **Trace panel** is displayed in the **Disassemble display mode**, a source-level search is performed on this tab, the subject data cannot be searched correctly. To perform a source-level search, make sure the Trace panel is displayed in the **Mixed display mode** or **Source display mode**.

Figure A-53. Trace Search Dialog Box: [Source Level] Tab



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Select the toolbar button  on the **Trace panel**.
- Choose [Find...] from the context menu on the **Trace panel**.

[Description of each area]**(1) [Search object] area**

Select the subject to search from the option buttons below.

The execution part is retrieved specifying the source line	Searches the specified source for an executed part (default). Only [Source and Line] is valid as the search condition.
The execution part is retrieved specifying the function	Searches the specified function for an executed part. Only [Function Name] is valid as the search condition.
The execution part is retrieved specifying the global variable	Searches the specified global variable for an accessed location. Only [Variable Name], [Kind] and [Value] are valid as the search condition.

(2) [Search condition] area**(a) [Source and Line]**

This item is enabled only when [The execution part is retrieved specifying the source line] is selected.

The character string specified here is searched from within the [Line/Address] area of the Trace panel. Enter a character string included in the source line to search directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-insensitive, and partial matches are also allowed.

Examples 1. main.c#40

2. main.c

3. main

(b) [Function Name]

This item is enabled only when [The execution part is retrieved specifying the function] is selected.

Enter a variable name to search directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-sensitive, and only perfect matches are allowed.

(c) [Variable Name]

This item is enabled only when [The execution part is retrieved specifying the global variable] is selected.

Enter a variable name to search directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

Searches are case-sensitive, and only perfect matches are allowed.

(d) [Kind]

This item is enabled only when [The execution part is retrieved specifying the global variable] is selected.

Select the type of access ([Reference/Substitution] (default), [Reference], or [Substitution]) from the drop-down list.

(e) [Value]

This item is enabled only when [The execution part is retrieved specifying the global variable] is selected.

Use a hexadecimal number to specify an accessed variable value.

Enter a variable value directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

The variable value can be specified as a range of values. In this case, enter variable values in both the right and left text boxes to specify a range.

If the right-side text box is blank, the fixed variable value specified in the left-side text box is used to perform a search of accessed location.

(3) [Search range] area

(a) [Number]

Specify a range of trace data to search by numbers displayed in the [Number] area of the [Trace panel](#).

Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "*last number*" are specified.)

Enter a number in decimal notation directly in the text box or select an input history item from the drop-down list (up to 10 history entries).

If the left-side text box is blank, the number "0" is assumed.

If the right-side text box is blank, the "*last number*" is assumed.

[Function buttons]

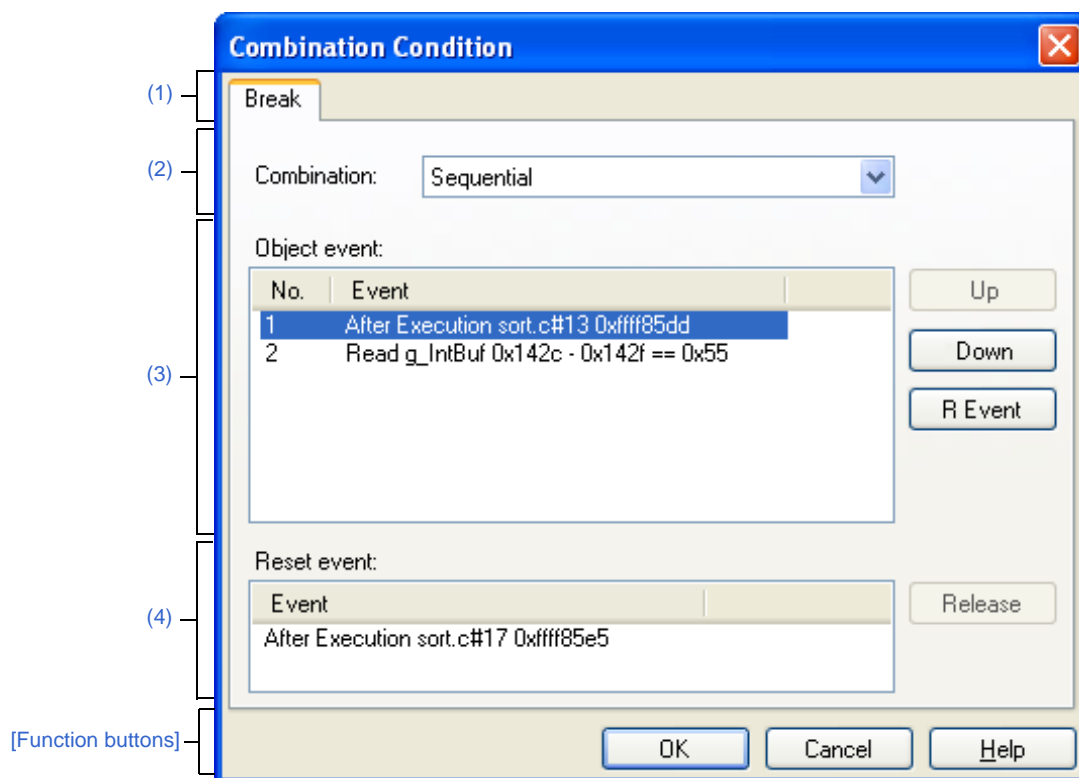
Button	Function
Search Backward	Performs a search in the direction toward smaller addresses within a specified range. The searched spot is put in selected state on the Trace panel . However, if an invalid value is specified, a message is displayed and a search of trace data is not performed. Also, if the trace panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.
Search Forward	Performs a search in the direction toward larger addresses within a specified range. The searched spot is put in selected state on the Trace panel . However, if an invalid value is specified, a message is displayed and a search of trace data is not performed. Also, if the trace panel is hidden, or if focus is moved to this dialog box while focus was present on another panel, this button is disabled.
Cancel	Nullifies the setting for a search of trace data and closes this dialog box.
Help	Displays help for this dialog box.

Combination Condition dialog box [E1][E20]

You can change the information on the combination break or the combination condition of trace event selected on the [Events panel](#).

Remark See "[2.17.7 Points to note regarding event setting](#)" for details on event setting.

Figure A-54. Combination Condition Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Events panel](#), move the caret to the combination break you wish to set, then select [Edit Condition ...] from the context menu.
- On the [Events panel](#), move the caret to the trace you wish to set, then select [Edit Condition ...] from the context menu.

[Description of each area]**(1) Tab selection area**

You can display detailed information on the matching combination conditions by the selection of a tab. Types of tabs to be displayed will differ depending on the event you have selected on the [Events panel](#).

Combination break	Only [Break] tab is displayed.
Trace	[Start Condition] tab and [Stop Condition] tab are displayed. By switching, you can set the combination condition for each tab.

(2) Combination condition selection area**(a) [Combination]**

Select the conditions from the following drop-down list.

Conditions	Functions
OR	The condition is satisfied when one of the set events is encountered.
AND	The condition is satisfied when all the set events are encountered irrespective of the time line.
Sequential	The condition is satisfied when the set events are encountered in the specified sequence,

- Cautions 1.** When the combination condition is "Sequential", break event (access type) can be specified for the 1st to the 3rd position.
- 2.** When the combination condition is "Sequential", break event (access type) for which address range condition is set can be specified only for the 1st position.
- 3.** For trace events, the combination condition available for a trace end event is only [OR].

(3) Object event condition display area**(a) Display of the list**

The object events to be combined are shown in the list.

No.	The events in the list are numbered from top to bottom. This item is displayed only when you have specified [Sequential] for the combination conditions. For the condition to be satisfied, events must be encountered in the order indicated by these numbers.
Event	Detailed information on event conditions is displayed. It is identical to the information displayed on the Events panel .

(b) Buttons

Buttons	Functions
Up	Moves the event serial number upward in the target event list. This button is enabled only when you have specified [Sequential] for the combination conditions.
Down	Moves the event serial number downward in the target event list. This button is enabled only when you have specified [Sequential] for the combination conditions.
R Event	Moves the event selected in the target event list to the reset event list. This button is enabled only when you have specified [Sequential] for the combination conditions.

(4) Reset event condition display/cancel area**(a) Display of the list**

Detailed conditions are displayed for the event which has been registered as a reset event. When the reset event displayed in this list is encountered, all the other event conditions which have been holding until then will be cleared.

Cautions 1. Only one reset event can be specified.

If you click [R Event] button with the reset event registered, the event selected in the target event list and the registered reset event will replace each other.

2. If you have changed the combination conditions from [Sequential] to [OR] or [AND] with the reset event registered, the event inside the reset event list cannot be included in the target event list. In order to make it a target event, you need to first cancel the reset event registration before making changes to the combination conditions.
3. For events in which a reset event is registered, you cannot specify a pass count condition.

(b) Button

Button	Function
Cancel	Moves the event inside the reset event list to the target event list. This button is enabled only when you have specified [Sequential] for the combination conditions.

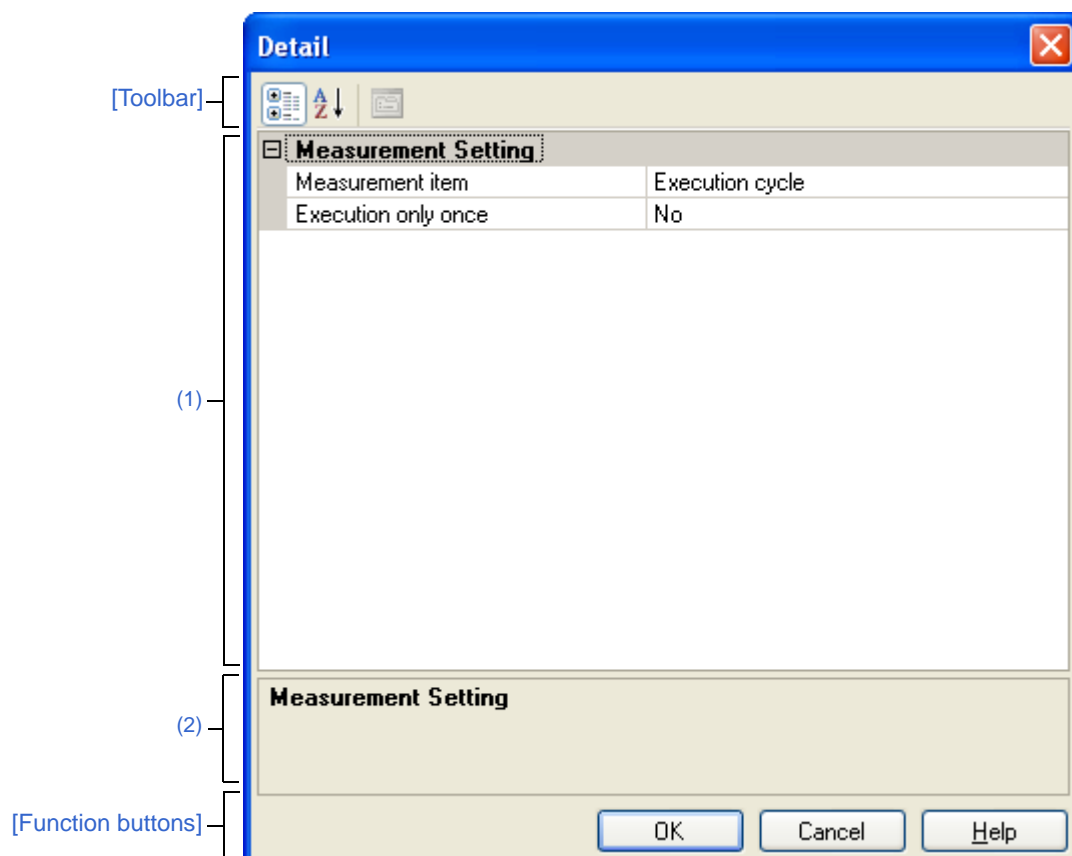
[Function buttons]

Buttons	Functions
OK	Applies the detailed settings specified in the dialog box to the combination break or to the trace and closes this dialog box.
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Detailed Settings of Timer Measurement dialog box [E1] [E20]

You can display and modify the detailed information on the timer event selected on the [Events panel](#). Note that you cannot edit the address value of the timer event in this dialog box. If you need to edit the address value, first delete the timer event and then create a new one. For details on timer event setting, see "[2.14 Measuring the Execution Time](#)".

Figure A-55. Detailed Settings of Timer Measurement Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[Description of each category\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Events panel](#), move the caret to the timer event you wish to set, then select [Edit Condition ...] from the context menu.



[Description of each area]**(1) Detailed information display/change area**

In this area, detailed information on the timer event selected in the [Events panel](#) is displayed by category in the list. Also, you can directly change its settings.

(2) Property description area

In this area, brief description of the categories and properties selected in the detailed information display/change area is displayed.

[Toolbar]

	Displays categories in the detailed information display/change area.
	Hides categories in the detailed information display/change area and rearranges only property items in the ascending order.

[Description of each category]**(1) [Measurement Setting]**

You can display and modify the detailed settings on timer measurement.

Measurement item	- RX600 Series Measurement items are specified.		
	- RX200 Series Execution cycle are displayed as measurement items.		
	Default	Execution cycle	
	Modifying	Select from the drop-down list	
Execution only once	Available values	Must be selected from the followings in the drop-down list. [RX600 Series] Execution cycle, Execution cycle (Supervisor mode), Exception and interrupt cycle, Exception cycle, Interrupt cycle, Execution count, Exception and interrupt count, Exception count, Interrupt count [RX200 Series] Execution cycle	
	Specifies whether to end the measurement after measuring the specified segment only once.		
	Default	No	
	Modifying	Select from the drop-down list	
	Available values	Yes	Ends the measurement after measuring the specified segment only once.
		No	The measurement value is cumulated every time you pass through the specified segment.

[Function buttons]

Buttons	Functions
OK	Applies the detailed settings specified in the dialog box to the timer event and closes this dialog box.
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Detailed Settings of Execution Events dialog box

This dialog box displays or changes detailed information on an execution-related event selected on the [Events panel](#).
Note that the execution-related events refer to the following events on the [Events panel](#).

- Hardware break
- Post-execution break in detailed information on combination break **[E1] [E20]**
- Post-execution event as start and end condition in detailed information on trace
- Post-execution event as start and end condition in detailed information on timer **[E1] [E20]**

Figure A-56. Detailed Settings of Execution Events Dialog Box [E1] [E20]

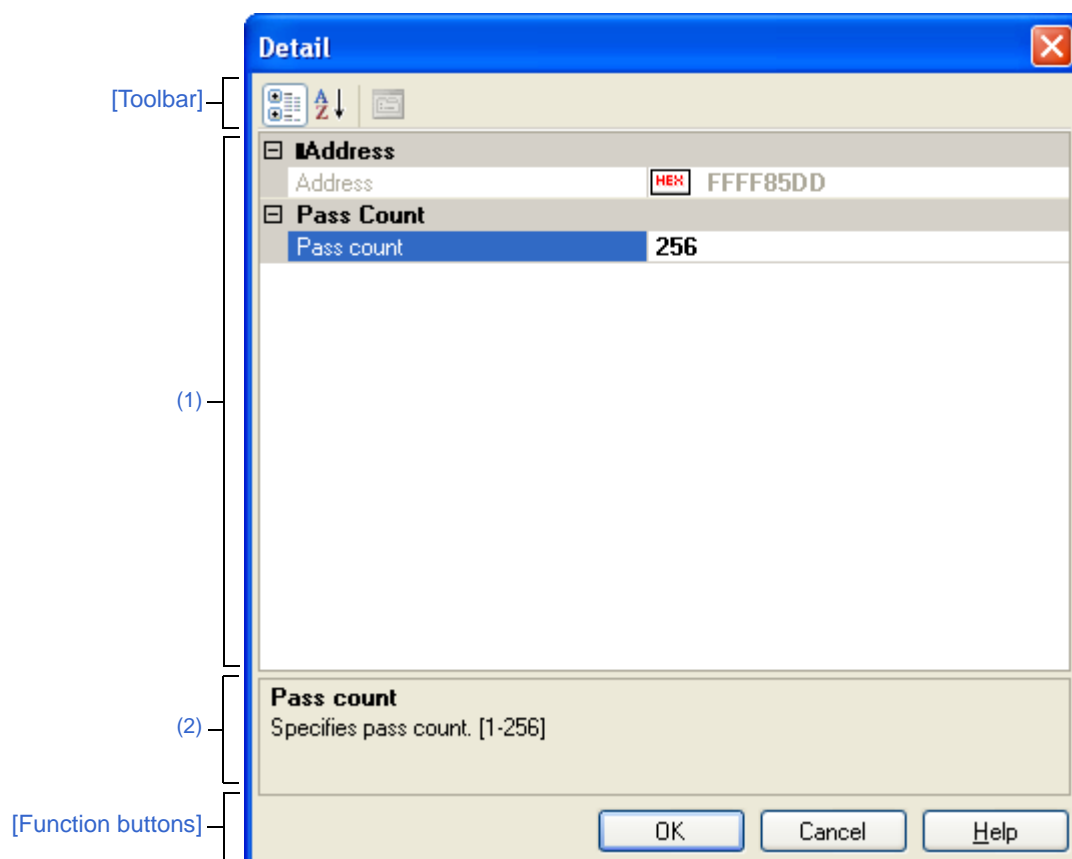
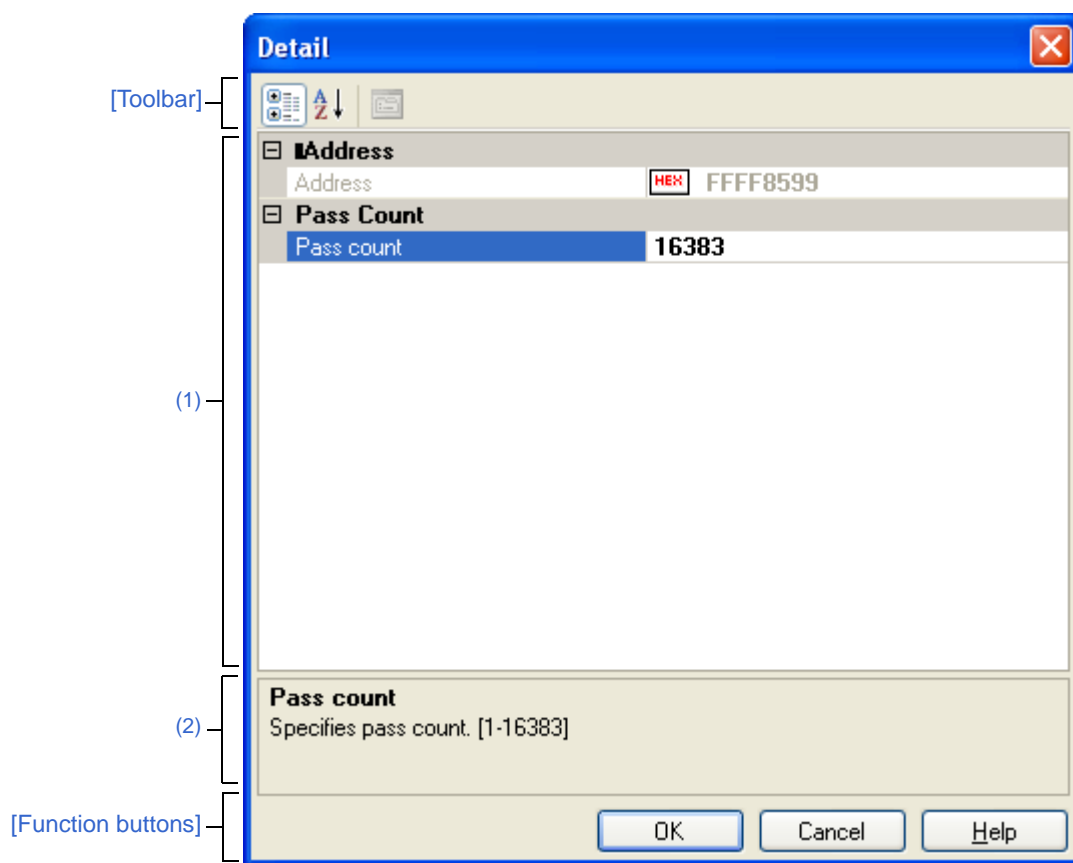


Figure A-57. Detailed Settings of Execution Events Dialog Box [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Description of each category]
- [Function buttons]

[How to open]

- On the [Events panel](#), move the caret to a hardware break you want to set and then select [Edit Condition ...] from the context menu.
- On the [Events panel](#), move the caret to an execution-related event in a trace that you want to set and then select [Edit Condition ...] from the context menu.

[E1] [E20]

- On the [Events panel](#), move the caret to an execution-related event in a combination break that you want to set and then select [Edit Condition ...] from the context menu.
- On the [Events panel](#), move the caret to an execution-related event in a timer result that you want to set and then select [Edit Condition ...] from the context menu.



[Description of each area]**(1) Detailed information display/change area**

This area displays, in list form of the category, the detailed information on an execution-related event that is selected on the [Events panel](#), allowing for its settings to be changed directly.

(2) Property description area

This area displays a simple description of the category or property selected in the detailed information display/change area.

[Toolbar]

	Displays a category in the detailed information display/change area.
	Hides categories in the detailed information display/change area and rearranges only property items in the ascending order.

[Description of each category]**(1) [Address]**

Address	Displays the address at which an execution-related event is set. No address values can be specified here.	
	Default	<i>Depends on selected event</i>
	modifying	Not changeable

(2) [Pass Count]

Pass count	Specify a pass count in decimal. The relevant event occurs when the event condition is met as many times as the entered pass count.	
	Default	1
	Modifying	By entering directly from the keyboard
	Available values	- [E1] [E20] 1 to 256 - [Simulator] 1 to 16,383

Cautions 1. [E1] [E20]

The pass count can be specified by any value other than 1 for only one event among all events (including access-related events).

If a pass count of other than 1 is specified for two events, an error results.

2. [E1(Serial) [RX200 Series]] [E20(Serial) [RX200 Series]]

No values other than 1 can be specified for the pass count.

[Function buttons]

Buttons	Functions
OK	Applies detailed settings made in this dialog box to execution-related events before closing the dialog box.
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Detailed Settings of Access Events dialog box

This dialog box displays or changes detailed information on an access-related event selected on the [Events panel](#).

Note that the access-related events refer to the following events on the [Events panel](#).

- Read, write, and read/write in detailed information on combination break **[E1] [E20]**
- Read, write, and read/write in detailed information on hardware break **[Simulator]**
- Read, write, and read/write in detailed information on trace
- Read, write, and read/write in detailed information on point trace
- Read, write, and read/write in detailed information on timer result **[E1] [E20]**

Figure A-58. Detailed Settings of Access Events Dialog Box [E1] [E20]

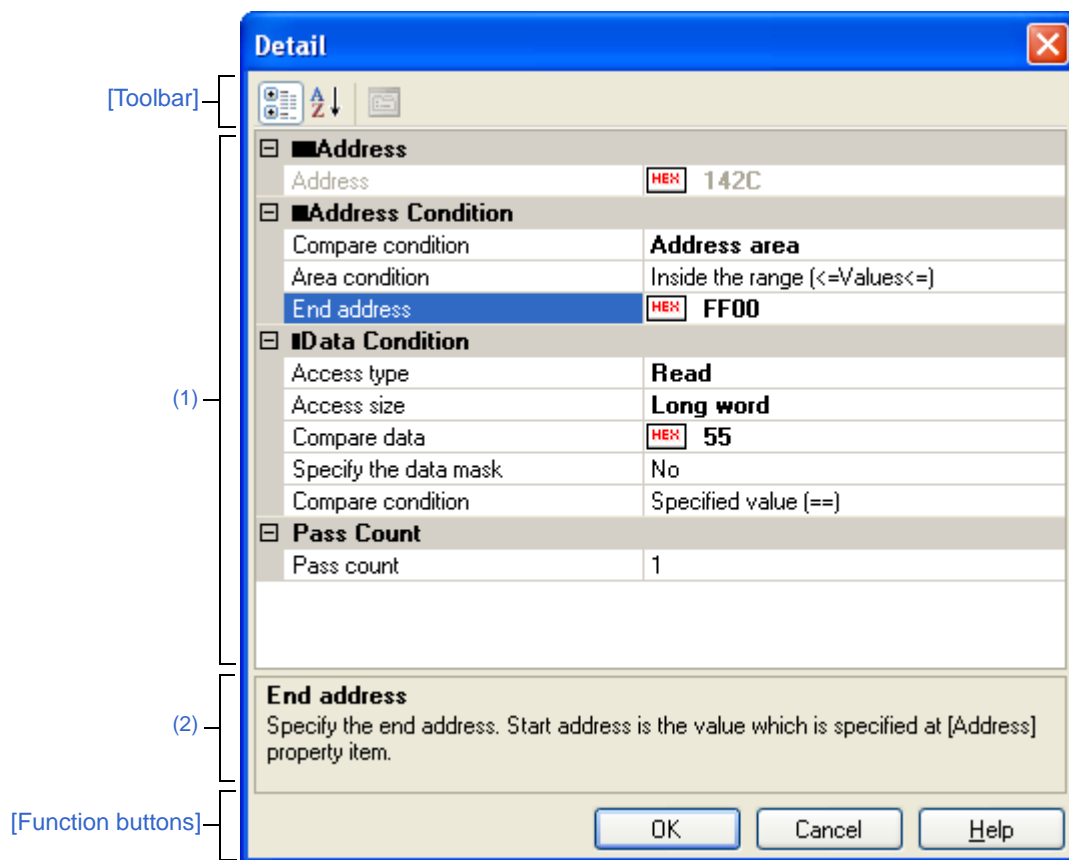
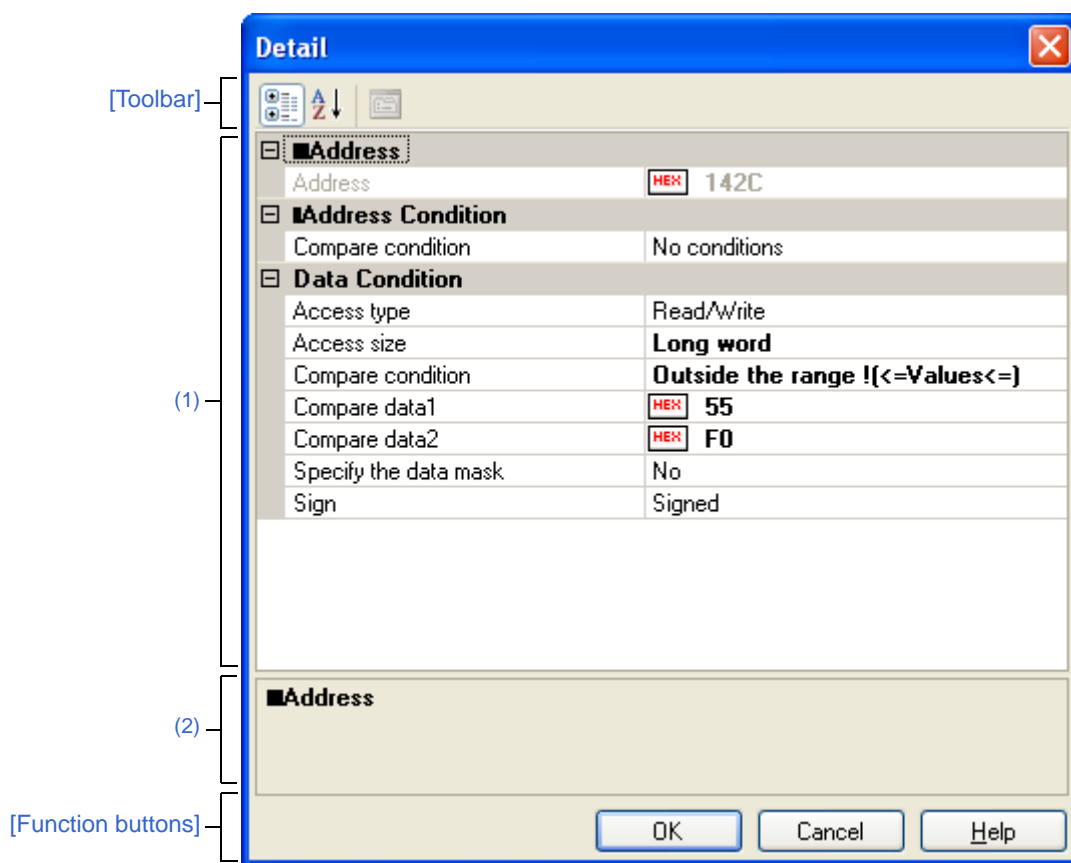


Figure A-59. Detailed Settings of Access Events Dialog Box [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Description of each category]
- [Function buttons]

[How to open]

- On the [Events panel](#), move the caret to an access-related event in a trace that you want to set and then select [Edit Condition ...] from the context menu.
- On the [Events panel](#), move the caret to an access-related event in a point trace that you want to set and then select [Edit Condition ...] from the context menu.

[E1] [E20]

- On the [Events panel](#), move the caret to an access-related event in a combination break that you want to set and then select [Edit Condition ...] from the context menu.
- On the [Events panel](#), move the caret to an access-related event in a timer result that you want to set and then select [Edit Condition ...] from the context menu.

[Simulator]

- On the [Events panel](#), move the caret to an access-related hardware break that you want to set and then select [Edit Condition ...] from the context menu.



[Description of each area]**(1) Detailed information display/change area**

This area displays, in list form of the category, the detailed information on an access-related event that is selected on the [Events panel](#), allowing for its settings to be changed directly.

(2) Property description area

This area displays a simple description of the category or property selected in the detailed information display/change area.

[Toolbar]

	Displays categories in the detailed information display/change area.
	Hides categories in the detailed information display/change area and rearranges only the property items in the ascending order.

[Description of each category]

The properties that can be displayed and changed differ with the debug tool used.

(1) For [\[E1\]](#) [\[E20\]](#)

(2) For [\[Simulator\]](#)

(1) For [\[E1\]](#) [\[E20\]](#)**(a) [Address]**

Address	Displays the address at which an access-related event is set. No address values can be specified here.	
	Default	<i>Depends on selected event</i>
	modifying	Not changeable

(b) [Address Condition]

Compare condition	Specify address compare condition.		
	Default	<i>No specification</i>	
	Modifying	By selecting from the drop-down list	
	Available values	No conditions	No address compare condition is specified
		Address area ^{Note1} Note2	Specify compare condition based on an address range
		Compare with address mask	Specify masked compare condition.

Area condition	Specify address comparison range condition. This property is displayed only when [Address Range] is selected in the [Compare condition] property.		
	Default	Inside the range (<=values<=)	
	Modifying	By selecting from the drop-down list	
	Available values	Inside the range (<=values<=)	Condition holds true for an access within the range
		Outside the range !(<=Values<=)	Condition holds true for an access outside the range
End Address	Specify the end address The start address is the value displayed in the [Address] property. This property is displayed only when [Address area] is selected in the [Compare condition] property.		
	Default	Blank	
	Modifying	By entering directly from the keyboard	
	Available values	0x0 to 0xFFFFFFFF	
	Address mask	Specify a mask value in hexadecimal. Note3 This property is displayed only when [Compare with address mask] is selected in the [Compare condition] property.	
Default		Blank	
Modifying		By entering directly from the keyboard	
Available values		0x0 to 0xFFFFFFFF	
Compare		Specify compare condition. This property is displayed only when [Compare with address mask] is selected in the [Compare condition] property.	
	Default	Specified value (==)	
	Modifying	By selecting from the drop-down list	
	Available values	Specified value (==)	True when matched with address compare condition.
		Any other value (!=)	True when not matched with address compare condition.

Notes 1. [RX200 Series]

Compare conditions based on an address area are not supported.

2. [RX600 Series]

You can specify "Address area" as the compare condition only to one event.

3. The address value to be used as the condition is masked bit-wise with a mask value for which "0" is specified.

Example) To set an address condition of 0x1000 to 0x1FFF

Address value : 0x1000

Mask value : 0xF000

(c) [Data Condition]

Access type	Specify the type of access.		
	Default	Read/Write	
	Modifying	By selecting from the drop-down list	
	Available values	Read	The specified type of access is a read.
		Write	The specified type of access is a write.
		Read/Write	The specified type of access is a read and a write.
Access size	Specify the access size.		
	Default	No conditions	
	Modifying	By selecting from the drop-down list	
	Available values	No conditions	No access size is specified. True for all access sizes.
		Byte	The specified access size is a byte.
		Word	The specified access size is a word.
		Long word	The specified access size is a long word.
Compare data	Specify compare data in hexadecimal.		
	Default	Blank	
	Modifying	By entering directly from the keyboard	
	Available values	0x0 to 0xFFFFFFFF	
Specify the data mask	Specify whether or not to specify a data mask.		
	Default	No	
	Modifying	By selecting from the drop-down list	
	Available values	Yes	Data mask is specified.
		No	No data mask is specified.
Mask value	Note Specify mask data in hexadecimal. This property is displayed only when [Yes] is specified in the [Specify the data mask] property.		
	Default	Blank	
	Modifying	By entering directly from the keyboard	
	Available values	0x0 to 0xFFFFFFFF	

Compare condition	Specify data compare condition. This property is displayed only when [Yes] is specified in the [Specify the data mask] property.		
	Default	Specified value (==)	
	Modifying	By selecting from the drop-down list	
	Available values	Specified value (==)	True when data compare condition matches.
		Any other value (!=)	True when data compare condition does not match.

Note The address value to be used as the condition is masked bit-wise with a mask value for which "0" is specified.

(d) [Pass Count]

Pass count	Specify a pass count in decimal. The relevant event occurs when the event condition is met as many times as the entered pass count.	
	Default	1
	Modifying	By entering directly from the keyboard
	Available values	1 to 256

Cautions 1. [E1] [E20]

The pass count can be specified by any value other than 1 for only one event among all events (including execution-related events).

If a pass count of other than 1 is specified for two events, an error results.

2. [E1(Serial) [RX200 Series]] [E20(Serial) [RX200 Series]]

The pass count can only be specified by 1. Any value other than 1 cannot be specified.

(2) For [Simulator]

(a) [Address]

Address	Displays the address at which an access-related event is set. No address values can be specified here.	
	Default	Depends on selected event
	Modifying	Not changeable

(b) [Address Condition]

Compare condition	Displays address compare condition.	
	Default	No specification
	Modifying	Not changeable

(c) Data Condition

Access type	Specify the type of access.		
	Default	<i>Depends on selected event</i>	
	Modifying	By selecting from the drop-down list	
	Available values	Read	The specified type of access is a read.
		Write	The specified type of access is a write.
		Read/Write	The specified type of access is a read and a write.
Access size	Specify the access size.		
	Default	<i>Depends on selected event</i>	
	Modifying	By selecting from the drop-down list	
	Available values	No conditions	No access size is specified. True for all access sizes.
		Byte	The specified access size is a byte.
		Word	The specified access size is a word.
		Long word	The specified access size is a long word.

Compare condition	Specify data compare condition. This property is not displayed when [No conditions] is specified in the [Access size] property.		
	Default	<i>Equal (==)</i>	
	Modifying	By selecting from the drop-down list	
	Available values	Equal (==)	True when data matches specified value.
		Not equal (!=)	True when data does not match specified value.
		Inverse sign	True when the sign is inverted between the previously accessed data and the data accessed this time. Note1
		Difference	True when the difference between the previously accessed data and the data accessed this time exceeds a specified value. Note1
		Greater than (>)	True when data is greater than specified value.
		Less than (<)	True when data is smaller than specified value.
		Greater than or equal to (>=)	True when data is equal to or greater than specified value.
		Less than or equal to (<=)	True when data is equal to or smaller than specified value.
		Inside the range (<=Values<=)	True when data is within the range of specified value. ([Compare data1] <= data <= [Compare data2])
		Outside the range !(<=Values<=)	True when data is outside the range of specified value. (data < [Compare data1] [Compare data2] < data)
No conditions		Compare data is not specified.	

Compare data1	Specify compare data in hexadecimal. This property is displayed when one of the following is specified in [Compare condition] property. <ul style="list-style-type: none">- Equal (==)- Not equal (!=)- Greater than (>)- Less than (<)- Greater than or equal to (>=)- Less than or equal to (<=)- Inside the range (<=Values<=)- Outside the range !(<=Values<=)		
	Default	Blank	
	Modifying	By entering directly from the keyboard	
	Available values	0x0 to 0xFFFFFFFF	
Compare data2	Specify compare data in hexadecimal. This property is displayed when one of the following is specified in [Compare condition] property. <ul style="list-style-type: none">- Inside the range (<=Values<=)- Outside the range !(<=Values<=)		
	Default	Blank	
	Modifying	By entering directly from the keyboard	
	Available values	0x0 to 0xFFFFFFFF	
Difference	Specify compare data in hexadecimal. This property is displayed when [Difference] is specified in [Compare condition] property.		
	Default	Blank	
	Modifying	By entering directly from the keyboard	
	Available values	0x0 to 0xFFFFFFFF	
Specify the data mask	Specify whether or not to specify a data mask. This property is displayed when one of the following is specified in [Compare condition] property. <ul style="list-style-type: none">- Equal (==)- Not equal (!=)- Greater than (>)- Less than (<)- Greater than or equal to (>=)- Less than or equal to (<=)- Inside the range (<=Values<=)- Outside the range !(<=Values<=)		
	Default	No	
	Modifying	By selecting from the drop-down list	
	Available values	Yes	A data mask is specified.
		No	No data mask is specified.

Mask value	Specify a mask value in hexadecimal. ^{Note2} This property is displayed when [Yes] is specified in [Specify the data mask] property.	
	Default	Blank
	Modifying	By entering directly from the keyboard
	Available values	0x0 to 0xFFFFFFFF
Sign	Specify whether or not the data to be compared includes a sign. This property is displayed when one of the following is specified in [Compare condition] property. <ul style="list-style-type: none"> - Difference - Greater than (>) - Less than (<) - Greater than or equal to (>=) - Less than or equal to (<=) - Inside the range (<=Values<=) - Outside the range !(<=Values<=) 	
	Default	Signed
	Modifying	By selecting from the drop-down list
	Available values	Signed Data is compared with the sign included.
		Unsigned Data is compared with no sign included.

- Notes 1.** For [Inverse sign] and [Difference], since comparison is made with the previous data, the condition never holds true after a reset and in the first determination of whether condition is true.
- 2.** The address value to be used as the condition is masked bit-wise with a mask value for which "0" is specified.

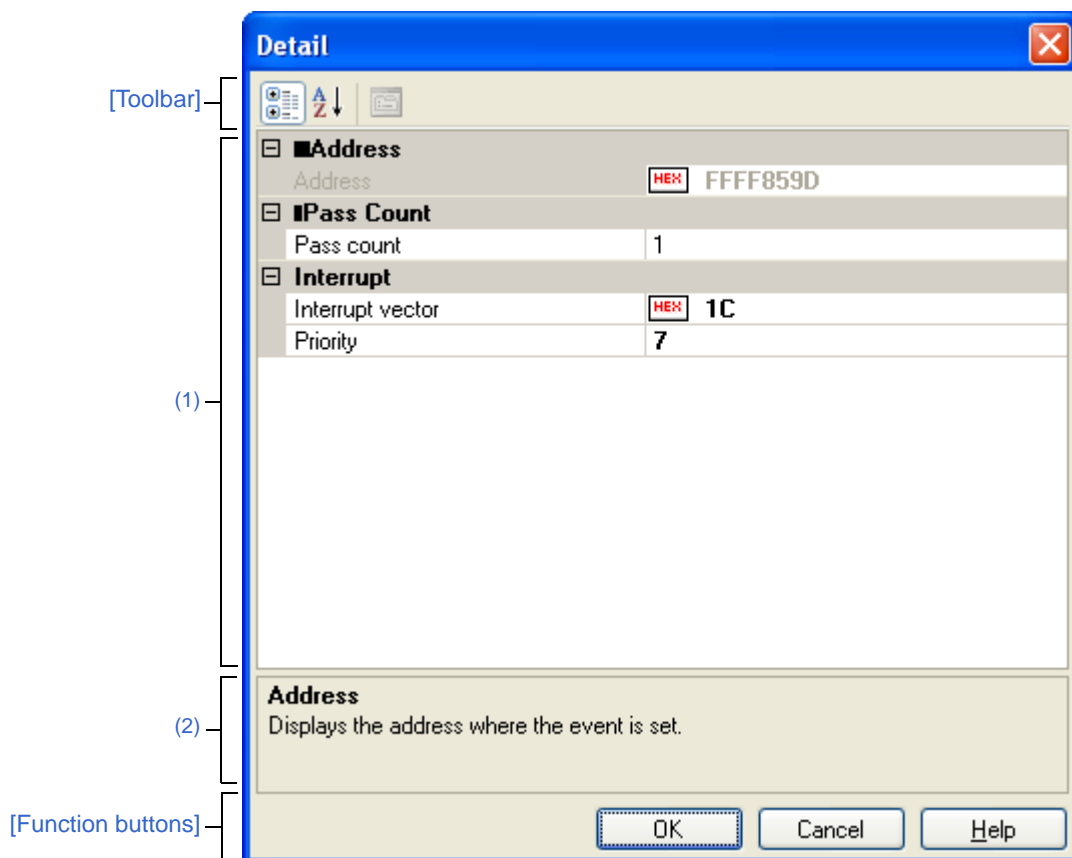
[Function buttons]

Buttons	Functions
OK	Applies the detailed settings made in this dialog box to the access-related events before closing the dialog box.
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Detailed Settings of Interrupt Events dialog box [Simulator]

This dialog displays or changes detailed information on an interrupt event selected on the [Events panel](#).

Figure A-60. Detailed Settings of Interrupt Events Dialog Box [Simulator]



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[Description of each category\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Events panel](#), move the caret to an interrupt event you want to set and then select [Edit Condition...] from the context menu.



[Description of each area]**(1) Detailed information display/change area**

This area displays, in list form of the category, the detailed information on an interrupt event that is selected on the [Events panel](#), allowing for its settings to be changed directly.

(2) Property description area

This area displays a simple description of the category or property selected in the detailed information display/change area.

[Toolbar]

	Displays a category in the detailed information display/change area.
	Hides categories in the detailed information display/change area and sorts only the property items in the ascending order.

[Description of each category]**(1) [Address]**

Address	Displays the address at which an interrupt event is set. No address values can be specified here.	
	Default	<i>Depends on the selected event</i>
	Modifying	Not changeable

(2) [Pass Count]

Pass count	Specifies a pass count in decimal. The relevant event occurs when the event condition is met as many times as the entered pass count.	
	Default	1
	Modifying	Direct entry from the keyboard
	Specifiable value	1 to 16,383

(3) [Interrupt]

Interrupt vector	Specifies an interrupt vector.	
	Default	<i>Depends on the selected event</i>
	Modifying	Direct entry from the keyboard
	Specifiable value	0 to 255

Priority level	Specifies an interrupt's priority level.	
	Default	<i>Depends on the selected event</i>
	Modifying	Direct entry from the keyboard
	Specifiable value	- [RX610 group] 1 to 8 If 8 is specified, the interrupt will act as a fast interrupt. - [Other than RX610 group] 1 to 16 If 16 is specified, the interrupt will act as a fast interrupt.

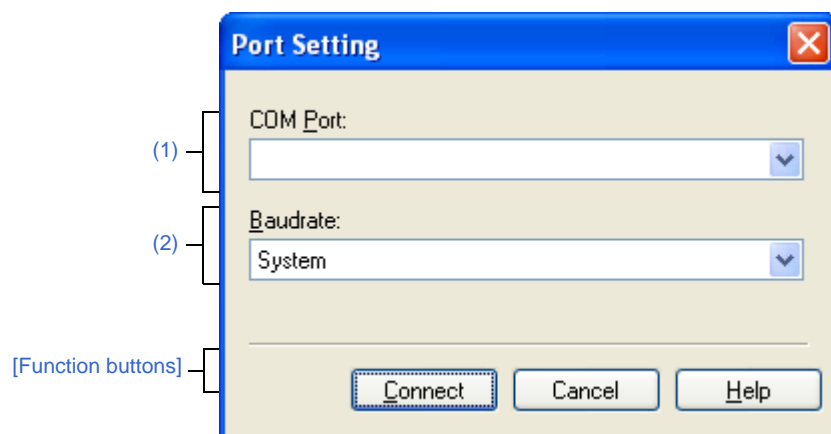
[Function buttons]

Buttons	Functions
OK	Applies the detailed settings made in this dialog box to interrupt events, and then closes the dialog box.
Cancel	Nullifies settings and closes this dialog box.
Help	Displays help for this dialog box.

Port Setting dialog box

This dialog box sets a COM port on the host machine to which communication from the microcontroller is redirected.

Figure A-61. Port Setting Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Debug Console panel](#), select [COM Port...] from the context menu.

[Description of each area]

(1) [COM Port] area

Select a COM port to which communication is redirected. If there are no COM ports available on the host machine, this drop-down list is blank.

(2) [Baudrate] area

Select a baud rate at which communication is performed with the redirected COM port.

The drop-down list displays the following baud rates. If "System" is selected, the baud rate set by the device manager applies.

System, 1200, 2400, 4800, 9600, 19200, 38400, 115200

[Function buttons]

Button	Function
Connect/Disconnect	Connects to (default) or disconnects from a COM port. When connected to a COM port, the button is labeled "Disconnect." When disconnected from a COM port, the button is labeled "Connect."
Cancel	Nullifies COM port settings and closes this dialog box.
Help	Displays help for this dialog box.

Scroll Range Settings dialog box

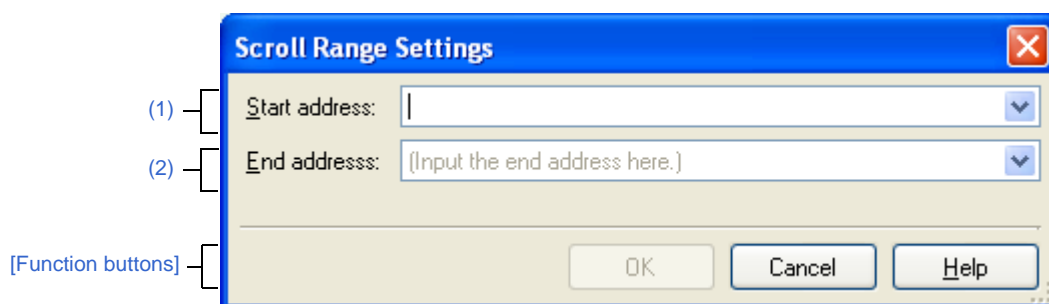
This dialog box is used to set the scroll range of the vertical scroll bar on the [Memory panel/Disassemble panel](#).

By setting the appropriate range, it is possible to improve the operability of a mouse (e.g. dragging) because the size of the vertical scroll bar on the panel is changed suitably.

Caution After setting a scroll range via this dialog box, the scroll range is not updated automatically even if the address evaluated by the address expression is changed because of such as a line assembly.

Remark It is possible to move outside the scroll range by using the [Page Up]/[Page Down]/[Up]/[Down] key, a button at either end of the scroll bar or a menu item related to the jump function.



Figure A-62. Scroll Range Settings Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Memory panel](#), click the  button on the toolbar.
- On the [Memory panel](#), select the [View] menu >> [Settings Scroll Range...] from the context menu.
- On the [Disassemble panel](#), click the  button on the toolbar.
- On the [Disassemble panel](#), select the [View] menu >> [Settings Scroll Range...] from the context menu.

[Description of each area]

(1) [Start address] area

Specify the start address of the range of scrolling.

You can either type an address expression directly into the text box (up to 1024 characters), or select it from the input history via the drop-down list (up to 10 items).

Note that the setting of the scroll range is not performed if "All" is selected in the drop-down list at this time (the scroll range is not limited).

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").

(2) [End address] area

Specify the end address of the range of scrolling.

You can either type an address expression directly into the text box (up to 1024 characters), or select it from the input history via the drop-down list (up to 10 items).

Note that this area becomes invalid if [Start address] is specified with [All].

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").

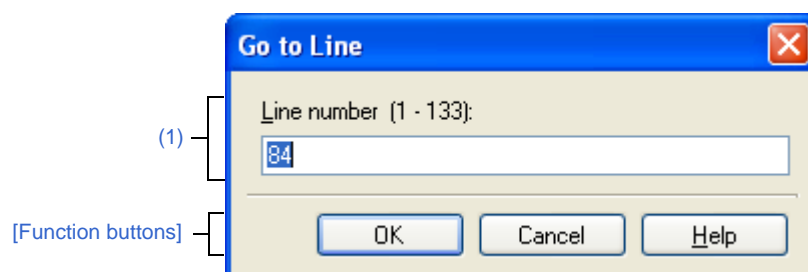
[Function buttons]

Button	Function
OK	Sets the specified scroll range for the target panel. Moves the caret to the start address, from the beginning of the area displayed in the target panel.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays help for this dialog box.

Go to Line dialog box

This dialog box is used to move the caret to a specified source line.

Figure A-63. Go to Line Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [Edit] menu, select [Go To...].
- On the [Editor panel](#), select [Go To...] from the context menu.

[Description of each area]

(1) [Line number (*valid line range*)] area

"(*valid line range*)" shows the range of valid lines in the current file.

Directly enter a decimal value as the number of the line you want to move the caret to.

You can also enter a symbol in this area.

By default, the number of the line where the caret is currently located in the [Editor panel](#) is displayed.

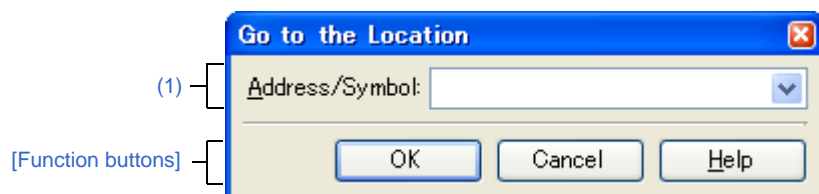
[Function buttons]

Button	Function
OK	Places the caret at the start of the specified source line.
Cancel	Cancels the jump and closes this dialog box.
Help	Displays help for this dialog box.

Go to the Location dialog box

This dialog box is used to move the caret to a specified position.

Figure A-64. Go to the Location Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- From the [Edit] menu, select [Go To...].
- On the [Disassemble panel](#), select [Go To...] from the context menu.
- On the [IOR panel](#), select [Go To...] from the context menu.

[Description of each area]

(1) [Address/Symbol] or [IOR] area

Specify the location to which the caret is jumped.

You can either type a location directly into the text box (up to 1024 characters), or select one from the input history via the drop-down list (up to 10 items).

The data to specify varies depending on the target panel, as follows:

Target Panel	Data Specified
Disassemble panel	Address expression
IOR panel	I/O register name

Remark If this dialog box is opened from the [Disassemble panel](#), By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "[2.20.2 Symbol name completion function](#)").

[Function buttons]

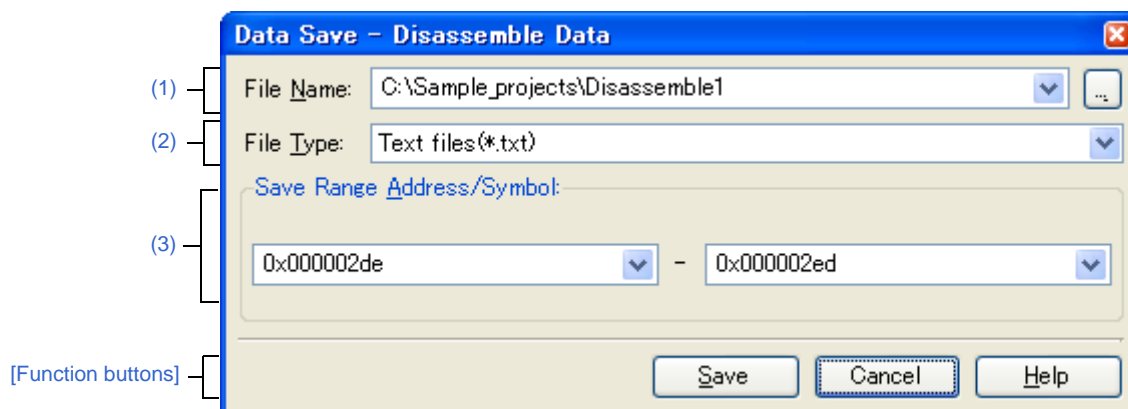
Button	Function
OK	Moves the caret to the specified location, from the beginning of the area displayed in the target panel.
Cancel	Cancels the jump and closes this dialog box.
Help	Displays help for this dialog box.

Data Save dialog box

This dialog saves the displayed contents of [Disassemble panel](#), [Memory panel](#) or [Trace panel](#) and the upload data.
(See "[2.5.3 Executing an upload](#)")

This dialog can only be opened when CubeSuite+ is connected with the debug tool.

Figure A-65. Data Save Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- While focus is on the [Disassemble panel](#), choose [Save Disassemble Data As...] from the [File] menu.
- While focus is on the [Memory panel](#), choose [Save Memory Data As...] from the [File] menu.
- While focus is on the [Trace panel](#), choose [Save Trace Data As...] from the [File] menu.
- Choose [Upload...] from the [Debug] menu.

[Description of each area]**(1) [File Name] area**

Specify a file name under which you want to save the data.

Enter a name directly in the text box (specifiable in up to 259 characters) or select an input history item from the drop-down list (up to 10 history entries).

It is also possible to select a file from the [Select Data Save File dialog box](#) that is opened by clicking the [...] button.

If you specify only a file name with no path information attached, the project folder is assumed.

(2) [File Type] area

Select the file format in which to save the data from the drop-down list below.

The selectable file format is determined by the type of data you are saving.

(a) When saving the displayed content of a panel

Text files (*.txt)	Text format (default)
CSV (Comma-Separated Variables) (*.csv)	CSV format ^{Note}

Note The data is saved with entries separated by commas (,).
If the data contains commas, each entry is surrounded by double quotes (") in order to avoid illegal formatting.

(b) When saving upload data

For the selectable file format, see "Table 2-3. Uploadable File Formats".

(3) [Save Range xxx] area

Specify the range of data to save.

You can either type ranges directly into the text boxes, or select them from the input history via the drop-down lists (up to 10 items).

The method of specifying the ranges will differ as follows depending on the type of data to be saved.

Type of Data	Description
Disassemble panel	Specify the range of addresses to save via the start and end addresses. Ranges can be entered as base-16 numbers or as address expressions. When a range is selected in the panel, that range is specified by default. When there is no selection, then the range currently visible in the panel is specified.
Memory panel	Specify the range of memory to save via the start and end addresses. Ranges can be entered as base-16 numbers or as address expressions. When a range is not selected in the panel, then the range currently visible in the panel is specified.
Trace panel	<ul style="list-style-type: none"> - Specifying a range to save Specify the trace range to save via the start and end trace numbers ^{Note}. Ranges can only be entered as base-10 numbers. - Saving all trace data From the drop-down list to the left, select [All Trace Data]. The text box to the right is disabled. All currently acquired trace data will be saved. The range currently visible in the panel is specified by default.
Upload data	Specify the range of memory to save via the start and end addresses. Ranges can be entered as base-16 numbers or as address expressions.

Note These are the numbers shown in the [Number] area of the Trace panel.

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.20.2 Symbol name completion function").

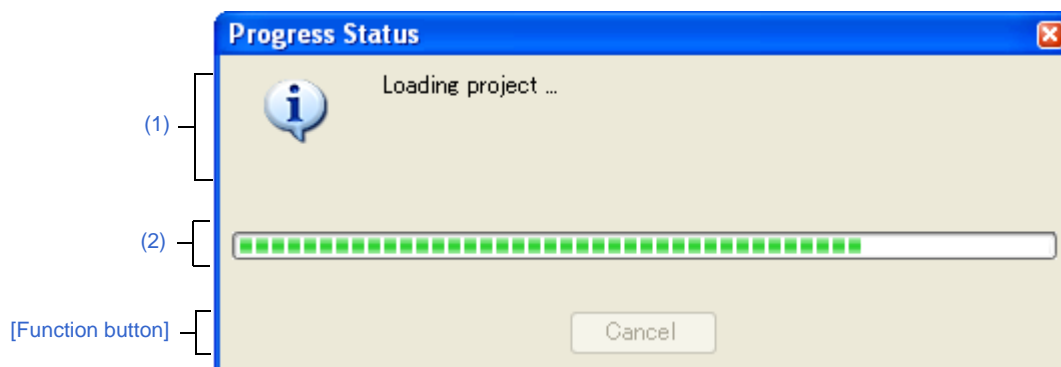
[Function buttons]

Button	Function
Save	Saves the data to a file with the specified filename, in the specified format.
Cancel	Cancels the save and closes this dialog box.
Help	Displays the help for this dialog box.

Progress Status dialog box

When CubeSuite+ is performing a process that takes time, it displays the progress of the process. This dialog box closes automatically when the process under execution is finished.

Figure A-66. Progress Status Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function button]

[How to open]

- Automatically displayed when a message is generated while a time-consuming process is underway.

[Description of each area]

(1) Message display area

Displays a message generated during a process (not editable).

(2) Progress bar

Shows the progress of the currently executed process by means of the length of a bar.

Note that when the progress rate reaches 100% (when the right edge of the bar is reached), this dialog closes automatically.

[Function button]

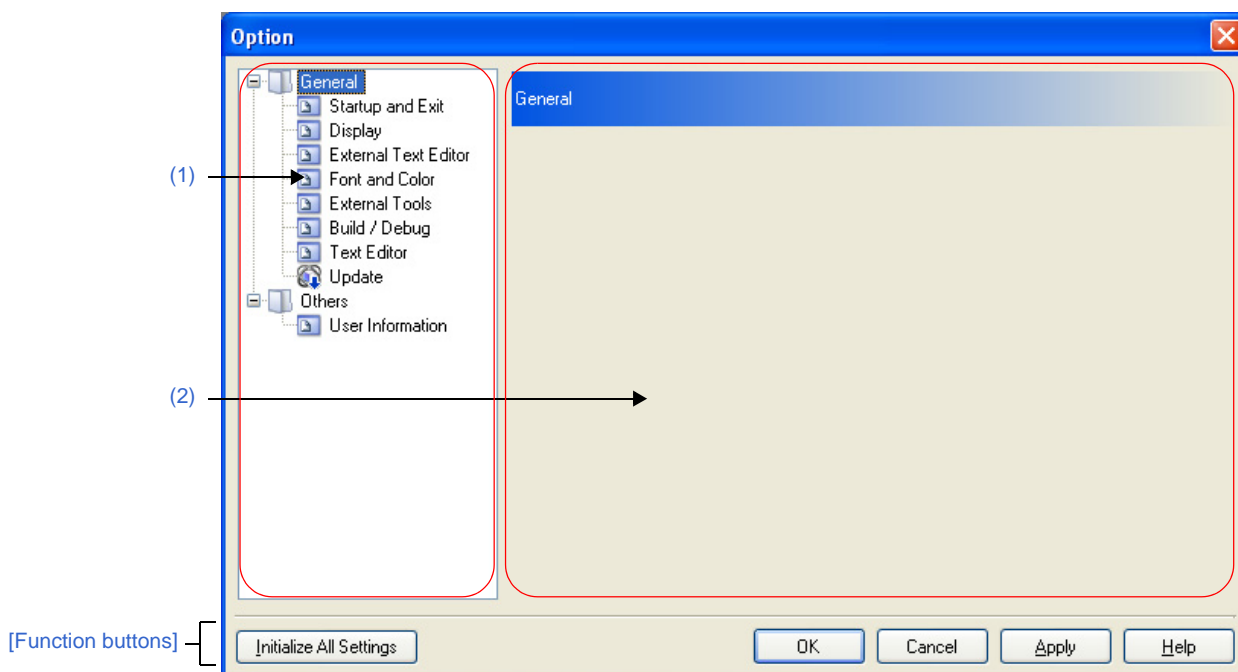
Button	Function
Cancel	Aborts the currently executed process and closes this dialog. However, if the process under execution cannot be aborted, this button is disabled.

Option dialog box

This dialog box makes various environment settings of CubeSuite+.

The settings in this dialog box are saved as settings for the current user of CubeSuite+.

Figure A-67. Option Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- Select [Options...] from the [Tool] menu.

[Description of each area]

(1) Category selection area

Select an item to set from the categories below.

Category	Content of Setting
[General - Startup and Exit] category	Make settings related to startup or exit time.
[General - Display] category	Make settings related to display.
[General - External Text Editor] category	Make settings related to external text editors.
[General - Font and Color] category	Make settings related to fonts and colors displayed on each panel.
[General - External Tools] category	Make settings with which external tools are started.
[General - Build/Debug] category	Make settings related to build or debug.
[General - Text Editor] category	Make settings related to text editors.

Category	Content of Setting
[General - Update] category	Make settings related to updates.
[Others - User Information] category	Make settings related to user information.

Remark For categories other than [General - Fonts and Colors] and [General - Build/Debug], see the separate edition, "CubeSuite+ Start."

(2) Setting area

This area is where various options for a selected category are set.

For details on how to set up each category, see the section in which the relevant category is described.

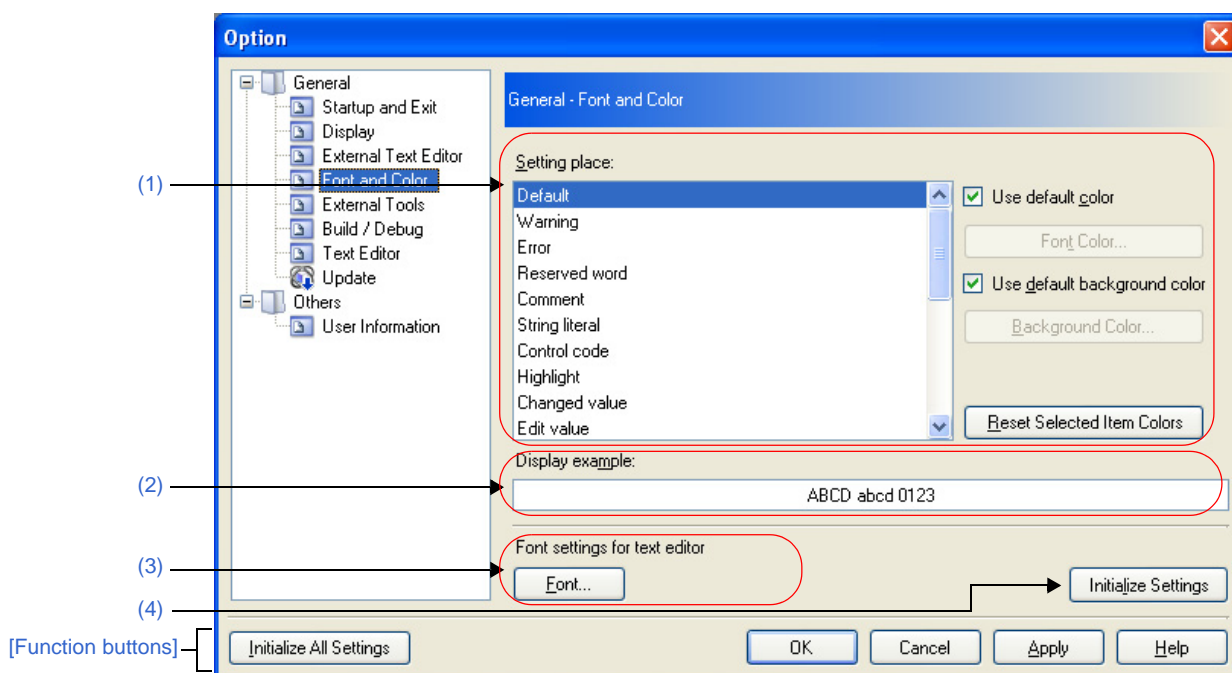
[Function buttons]

Button	Function
Initialize All Settings	Restores all of the set items in this dialog box to their default state. For the [General - External Tools] category, however, newly registered contents are not deleted.
OK	Applies changes and closes this dialog box.
Cancel	Nullifies changes and closes this dialog box.
Apply	Applies changes without closing this dialog box.
Help	Displays help for this dialog box.

[General - Font and Color] category

In this category, you can configure fonts and colors displayed on each panel.

Figure A-68. Option Dialog Box ([General - Font and Color] Category)



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Select [Options...] from the [Tool] menu.

[Description of each area]**(1) Color setting area**

This area is where colors are set.

(a) [Setting place] area

Select from the list a spot whose display color is to be specified.

The relationship between the items displayed in the list and the colors set for each by default is as follows.

Item	Example Display			Description
Default ^{Note}	AaBbCc	Text color	Black	Standard display colors on all windows and panels
		Background color	White	

Item	Example Display			Description
Warning	AaBbCc	Text color	Blue	Display colors of warning messages on the Output panel , as well as display colors for file names with "warnings included" on the Project Tree panel
		Background color	Standard color	
Error	AaBbCc	Text color	Red	Display colors of error messages on the Output panel , as well as display colors for file names with "errors included" on the Project Tree panel
		Background color	Light gray	
Reserved word	AaBbCc	Text color	Brown	Display colors of reserved words on the Editor panel for compilers/assemblers used
		Background color	Standard color	
Comment	AaBbCc	Text color	Green	Display colors of comment parts (for C source files, "/* to */") on the Editor panel
		Background color	Standard color	
String literal	AaBbCc	Text color	Gray	Display colors of string literals on the Editor panel
		Background color	Standard color	
Control code	AaBbCc	Text color	Blue-green	Display colors of control characters on the Output panel
		Background color	Standard color	
Highlight	AaBbCc	Text color	White	Display colors of highlighted spots in plug-in products, etc.
		Background color	Reddish purple	
Changed value	AaBbCc	Text color	Light brown	Display colors on the Memory panel , CPU Register panel , Local Variables panel , IOR panel and Watch panel of spots whose values have been changed by program execution
		Background color	Cream	
Edit value	AaBbCc	Text color	Blue	Display colors on the Memory panel , CPU Register panel , Local Variables panel , IOR panel and Watch panel of spots whose values have been forcibly changed by user
		Background color	Standard color	
Current PC	AaBbCc	Text color	Black	Display colors on the Editor panel and Disassemble panel of a line where the current PC position exists
		Background color	Gold	
Breakpoint	AaBbCc	Text color	Black	Display colors on the Editor panel and Disassemble panel of lines where breakpoints are set
		Background color	Salmon pink	
Update periodic	AaBbCc	Text color	Pink	Display colors on the Memory panel and Watch panel of areas whose display is set to be updated in real time
		Background color	Standard color	
Read or fetch	AaBbCc	Text color	Standard color	Display colors on the Memory panel and Trace panel of spots that have been read or fetched
		Background color	Light green	

Item	Example Display			Description
Write		Text color	Standard color	Display colors on the Memory panel and Trace panel of spots that have been written
		Background color	Orange	
Read and write		Text color	Standard color	Display colors on the Memory panel and Trace panel of spots that have been read and written
		Background color	Light blue	
Lost		Text color	Standard color	Display colors on the Memory panel of spots whose values obtained from the debug tool are incorrect
		Background color	Light gray	
Coverage 100%		Text color	Standard color	Display colors on the Editor panel and Disassemble panel of lines whose code coverage rates are 100%
		Background color	Light green	
Coverage 1 - 99%		Text color	Standard color	Display colors on the Editor panel and Disassemble panel of lines whose code coverage rates are 1 to 99%
		Background color	Light pink	
Coverage 0%		Text color	Standard color	Display colors on the Editor panel and Disassemble panel of lines whose code coverage rates are 0% (unexecuted)
		Background color	Light gray	
Invalid		Text color	Gray	Display colors on the Memory panel of areas that are not memory-mapped, and of file names that are not actually present on the Project Tree panel
		Background color	Standard color	

Note The [Default] text and background colors depend on how Windows is set on the host machine used. Here, the default settings of Windows, namely "text color: black" and "background color: white," are shown.

(b) [Use default color]

<input checked="" type="checkbox"/>	The items selected in the [Setting place] area are displayed using the standard text color.
<input type="checkbox"/>	Specify any color for the text color of the items selected in the [Setting place] area . The [Font Color...] button is enabled.

(c) [Use default background color]

<input checked="" type="checkbox"/>	The items selected in the [Setting place] area are displayed using the standard background color.
<input type="checkbox"/>	Specify any color for the background color of the items selected in the [Setting place] area . The [Background Color...] button is enabled.

(d) Buttons

Font Color...	Opens the Edit Colors Dialog Box , making it possible to specify a text color for the item selected in the [Setting place] area . However, this button is disabled when the [Use default color] checkbox is checked.
Background Color...	Opens the Edit Colors Dialog Box , making it possible to specify a background color for the item selected in the [Setting place] area . However, this button is disabled when the [Use default background color] checkbox is checked.
Reset Selected Item Colors	Resets color information for the item selected in the [Setting place] area to restore it to its default.

Figure A-69. Edit Colors Dialog Box**(2) [Display example] area**

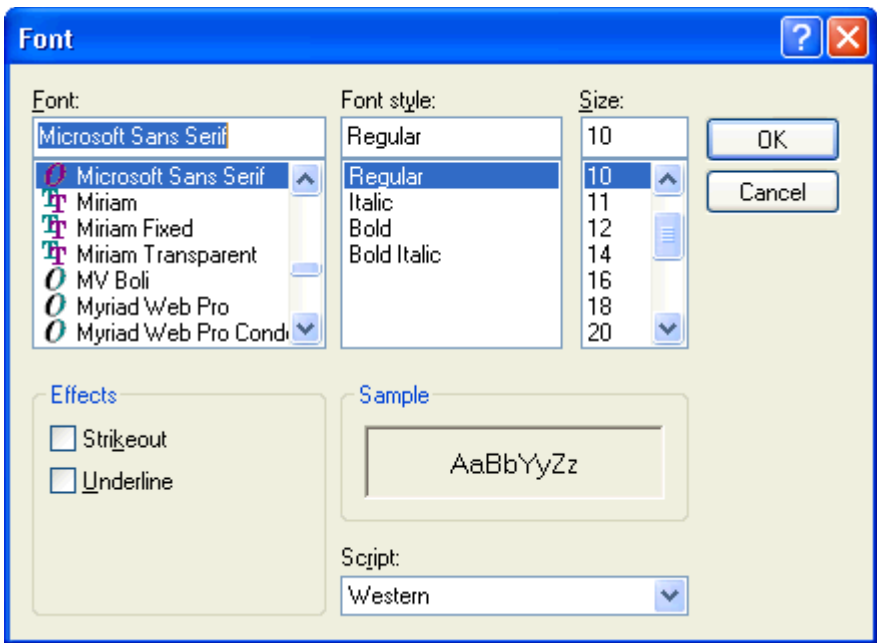
Shows an example of how the colors and fonts specified in the [Color setting area](#) and the [\[Font settings for text editor\] area](#) will be displayed.

Although this example shows a character string "AaBbCc" by default, any character string can be entered directly in the text box.

(3) [Font settings for text editor] area

Clicking the [Font...] button will open the [Font Dialog Box](#) below in which you can select the font to be used by the text editor.

Figure A-70. Font Dialog Box



(4) Buttons area

Initialize Settings	Restores specifications for all of the currently displayed items to their default.
---------------------	--

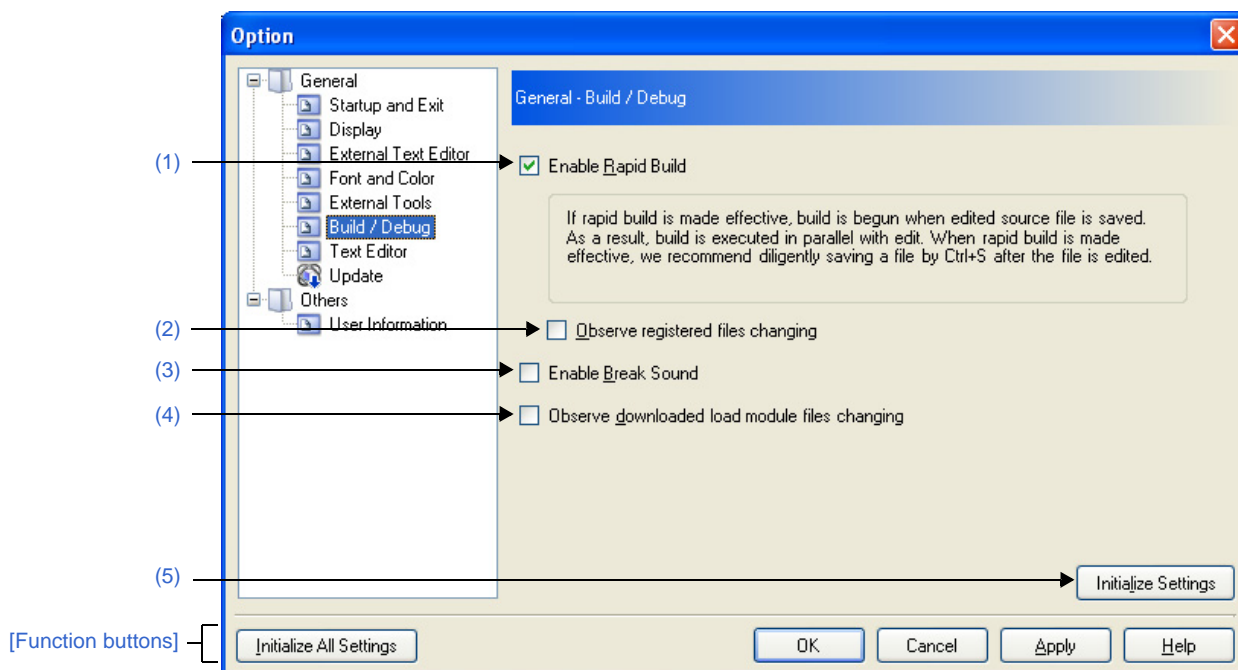
[Function buttons]

Button	Function
Initialize All Settings	Restores all of the set items in this dialog box to their default state. For the [General - External Tools] category, however, newly registered contents are not deleted.
OK	Applies changes and closes this dialog box.
Cancel	Nullifies changes and closes this dialog box.
Apply	Applies changes without closing this dialog box.
Help	Displays help for this dialog box.

[General - Build/Debug] category

In this category, you can configure settings related to build and debug.

Figure A-71. Option Dialog Box ([General - Build/Debug] Category)



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Select [Options...] from the [Tool] menu.

[Description of each area]**(1) [Enable Rapid Build]**

<input checked="" type="checkbox"/>	Enable the rapid build function ^{Note} (default).
<input type="checkbox"/>	Do not use the rapid build function.

Note This function automatically starts a build each time you save the edited source file.
By enabling this function, it is possible to edit source files and build the project at the same time.
When using this function, we recommend that you save frequently after editing the source file.

(2) [Observe registered files changing]

<input checked="" type="checkbox"/>	Monitor the source files registered in the project for any changes and start a rapid build when the files are edited and saved by an external editor, etc.
<input type="checkbox"/>	Do not monitor the source files registered in the project for any changes. Do not start a rapid build when the files are edited and saved by an external editor, etc (default).

Remark This feature is enabled only when the [\[Enable Rapid Build\]](#) checkbox is checked.

Caution In case any nonexistent source file (one grayed out in the project tree) is registered in the project, checking this box will not start the monitoring even when the said file is registered again by the Explorer, etc. To activate the monitoring mode, you need to either reload the project file or check this box again after unchecking it and closing this dialog box.

(3) [Enable Break Sound]

<input checked="" type="checkbox"/>	Beep when program execution is halted by a break event (hardware or software break).
<input type="checkbox"/>	Do not beep when program execution is halted by a break event (hardware or software break) (default).

(4) [Observe downloaded load module files changing]

<input checked="" type="checkbox"/>	Monitor the load module files downloaded to the debug tool for any changes, and display a message, when changes are detected, to confirm whether to perform the download.
<input type="checkbox"/>	Do not monitor the load module files downloaded to the debug tool for any changes (default).

(5) Buttons area

Initialize Settings	Restores all of the currently displayed items to their default state.
---------------------	---

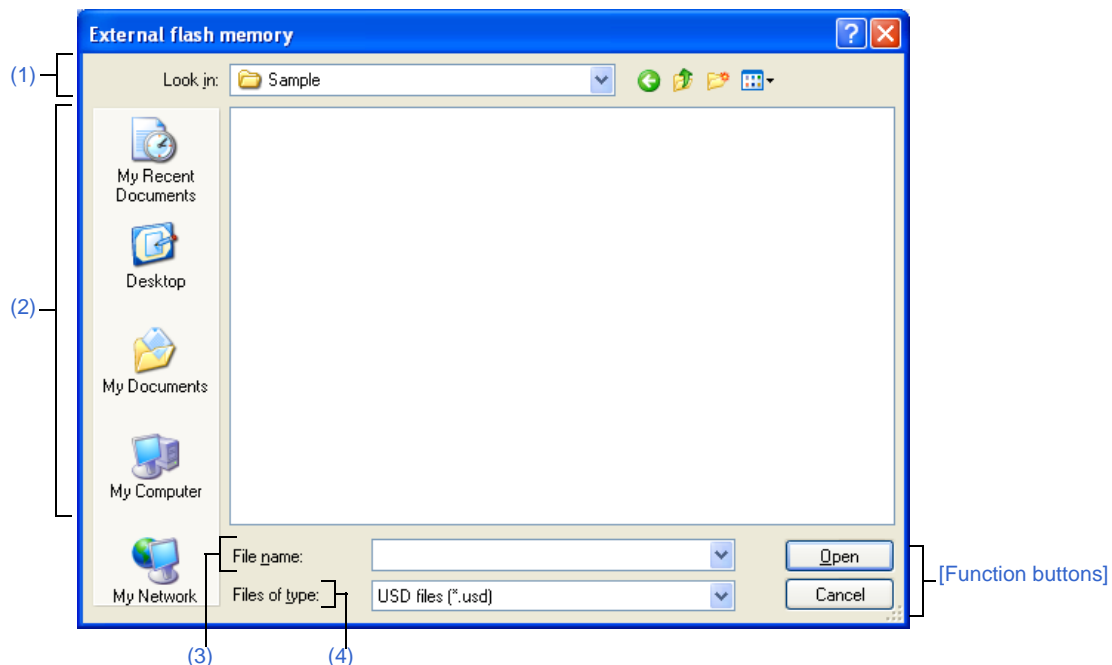
[Function buttons]

Button	Function
Initialize All Settings	Restores all the settings in this dialog box to their default state. In the [General - External Tools] category, however, newly registered contents are not deleted.
OK	Applies changes and closes this dialog box.
Cancel	Nullifies changes and closes this dialog box.
Apply	Applies changes without closing this dialog box.
Help	Displays help for this dialog box.

External flash memory dialog box [E1] [E20]

In this dialog box, you can select the external flash definition file (USD file) to be registered.

Figure A-72. External flash memory Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- In the [\[External Flash\] \[E1\] \[E20\]](#) category provided in the [\[Connect Settings\]](#) tab on the [Property panel](#), click [...] button which is displayed by selecting [\[File\]](#) property in the [\[External flash definition file\]](#) property.

[Description of each area]

(1) [Look in] area

Selects the folder in which a file to download is stored from the drop-down list.

(2) File list area

Display the list of files that match the conditions specified in [\[Look in\]](#) and [\[Files of type\]](#) areas.

(3) [File name] area

Specifies the file name to be registered.

(4) [Files of type] area

Selects the file type to be registered from the following drop-down list.

USD files (*.usd)	User system definition file format (default)
All files (*.*)	All file formats

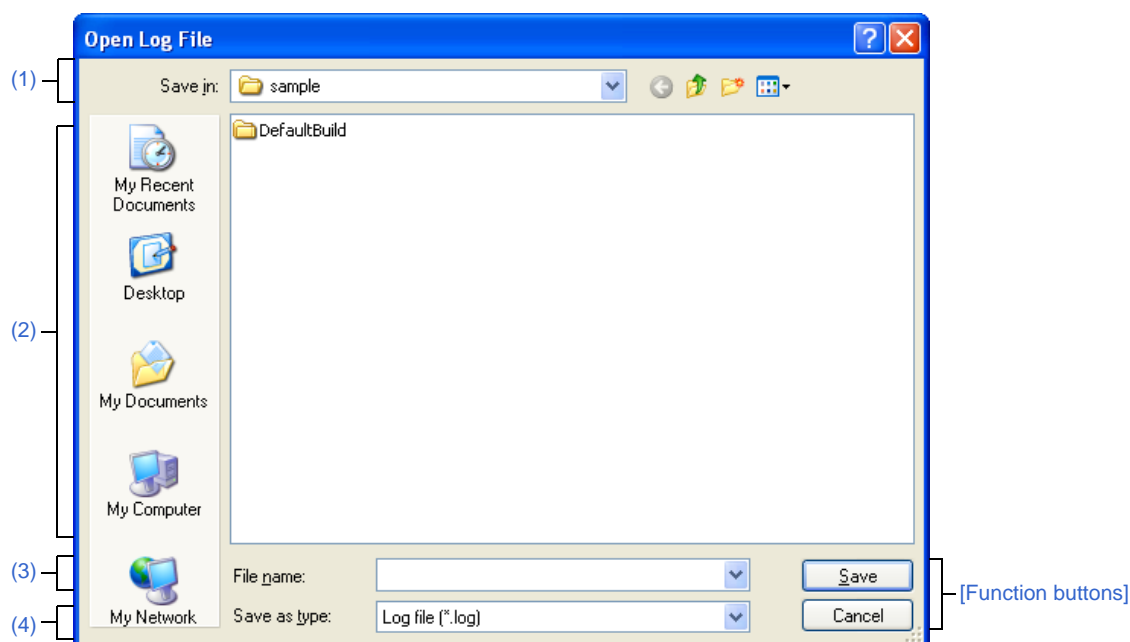
[Function buttons]

Button	Function
Open	Adds the specified file to the [File] property in the [External flash definition file] property.
Cancel	Closes this dialog box.

Open Log File dialog box

This dialog box selects a log file in which to save the displayed content of the [Debug Console panel](#).

Figure A-73. Open Log File Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- On the [Debug Console panel](#), select [Log File...] from the context menu.

[Description of each area]

(1) [Save in] area

Select a folder in which to save a log file.

(2) List of files area

This area displays a list of files that match the conditions selected in the [Save in] area and [Save as type] area.

(3) [File name] area

Specify a file name under which data is saved.

(4) [Save as type] area

This area displays the following types of files (file types).

Log file (*.log)	Text format (default)
All file (*.*)	All the formats

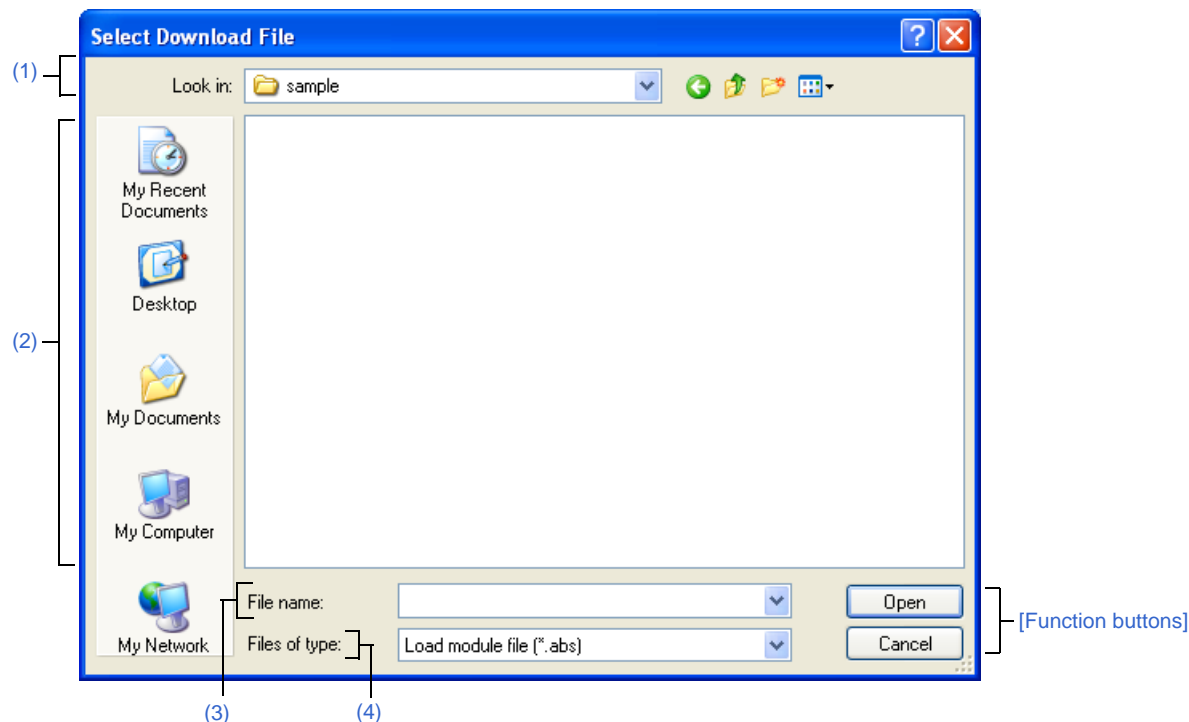
[Function buttons]

Button	Function
Save	Saves a file under a specified file name.
Cancel	Closes this dialog box.

Select Download File dialog box

This dialog box selects a file to download.

Figure A-74. Select Download File Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- Click the [...] button in the [File] item in the [Download file property] area of the [Download Files dialog box](#).

[Description of each area]

(1) [Look in] area

From the drop-down list, select the folder which contains the file you wish to download.

(2) File list area

This area displays a list of files that match the conditions selected in the [Look in] and [Files of type] areas.

(3) [File name] area

Specify the name of a file you wish to download.

(4) [Files of type] area

Select the type of a file to download (file type) from the drop-down list below.

Load module file (*.abs)	Load module format (default)
Hex file (*.hex)	Hex format
S-Record file (*.mot)	S-Record format
Binary data file (*.bin)	Binary data format
All files (*.*)	All file formats

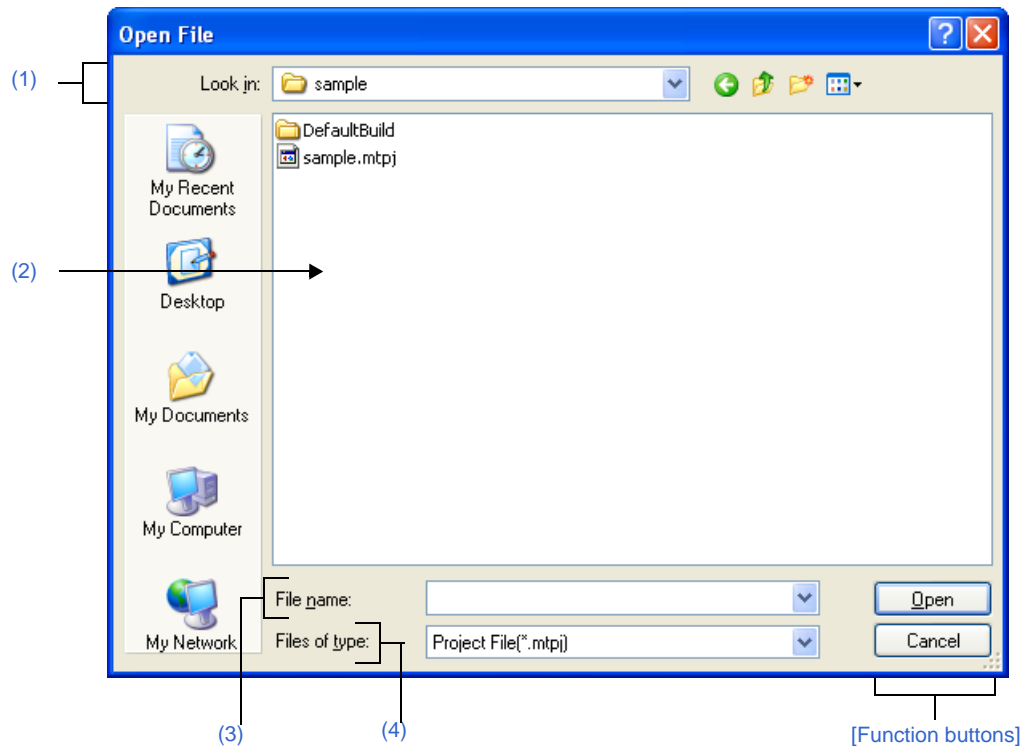
[Function buttons]

Button	Function
Open	Adds a specified file to the Download Files dialog box .
Cancel	Closes this dialog box.

Open File dialog box

You can select files you want to open in this dialog box.

Figure A-75. Open File Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [File] menu, select [Open File...] or [Open with Encoding...].

[Description of each area]

(1) [Look in] area

Select the folder which contains the file you want to open.

When you first open this dialog box, "C:\Documents and Settings \user-name\My Documents" will be selected.

After the second time, the last selected folder will be shown by default.

(2) File List area

This area shows the list of files that meet the conditions specified in [Look in] and [Files of type] areas.

(3) [File name] area

Specify the name of the file to be opened.

(4) [Files of type] area

Select the type of the file to be opened.

All files (*.*)	All formats
Project File(*.mtpj)	Project file
Project File for CubeSuite(*.cspj)	Project file for CubeSuite
Workspace File for HEW(*.hws)	Workspace file for HEW
Project File for HEW(*.hwp)	Project file for HEW
Workspace File for PM+(*.prw)	Workspace file for PM+
Project File for PM+(*.prj)	Project file for PM+
C source file (*.c)	C language source file
C++ source file (*.cpp; *.cc; *.cp)	C++ language source file
Header file (*.h; *.hpp; *.inc)	Header file
Assemble file (*.src)	Assembler source file
Map file (*.map; *.lbp)	Map file
Hex file (*.hex)	Hex file
Assemble list file (*.lst)	Assemble list file
S record file (*.mot)	S record file
Text file (*.txt)	Text format

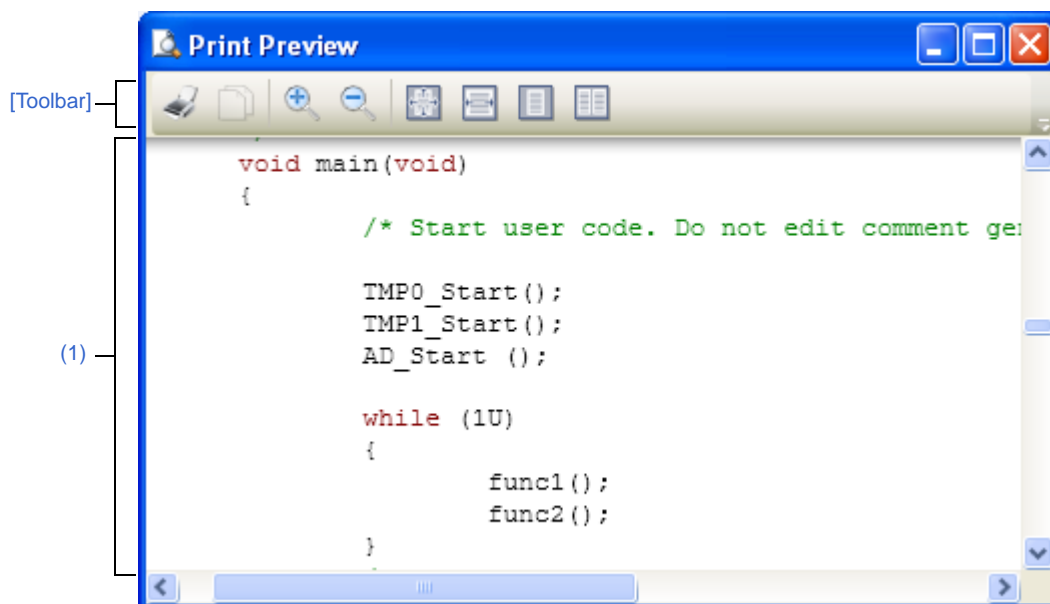
[Function buttons]

Button	Function
Open	<ul style="list-style-type: none"> - When you select [Open File...] from the [File] menu: The selected file will open. - When you select [Open with Encoding...] from the [File] menu: Encoding dialog box will open.
Cancel	This dialog box will close.

Print Preview window

This window is used to preview the source file before printing.

Figure A-76. Print Preview Window



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Toolbar\]](#)
- [\[Context menu\]](#)

[How to open]

- Focus the [Editor panel](#), and then select [Print Preview] from the [File] menu.

[Description of each area]**(1) Preview area**

This window displays a form showing a preview of how and what is printed.

[Toolbar]

	Opens the Print dialog box provided by Windows to print the current Editor panel as shown by the print preview form.
	Copies the selection into the clipboard.
	Increases the size of the content.
	Decreases the size of the content.
	Displays the preview at 100-percent zoom (default).
	Fits the preview to the width of this window.
	Displays the whole page.
	Displays facing pages.

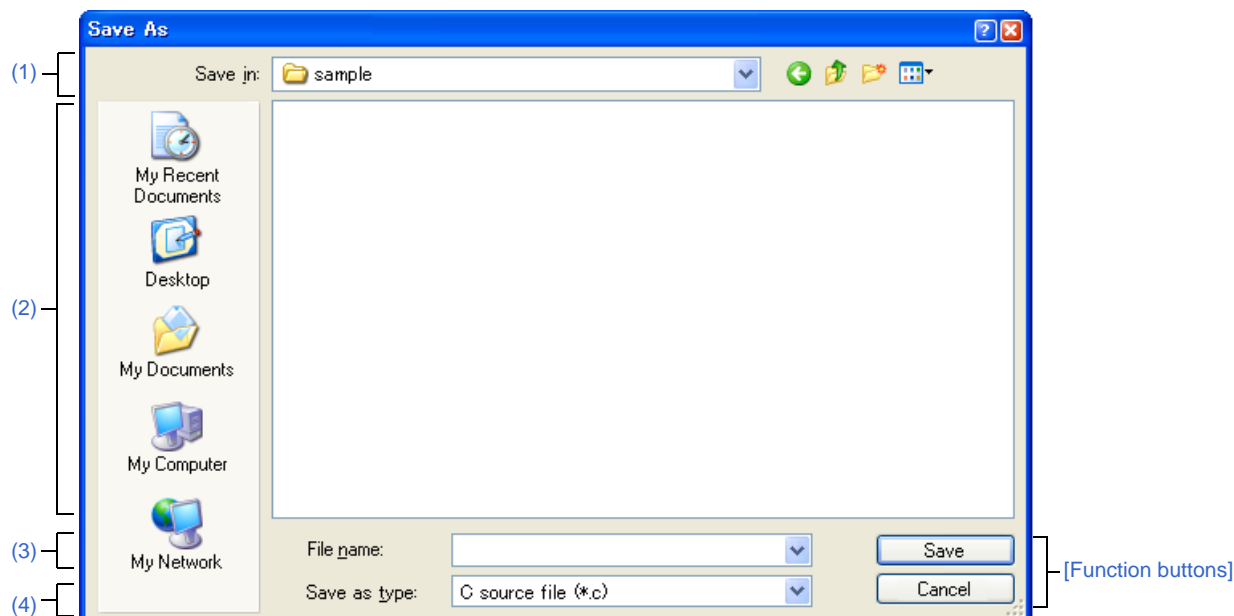
[Context menu]

Increase Zoom	Increases the size of the content.
Decrease Zoom	Decreases the size of the content.

Save As dialog box

Saves the file being edited under a new name or saves the content of any panel in a file with a specified name.

Figure A-77. Save As Dialog Box



Here, the following items are explained.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- While focus is present on [Editor panel](#), chose [\[Save File Name As...\]](#) from the [\[File\]](#) menu.
- While focus is present on [CPU Register panel](#), choose [\[Save CPU Register Data As...\]](#) from the [\[File\]](#) menu.
- While focus is present on [Watch panel](#), choose [\[Save Watch Data As...\]](#) from the [\[File\]](#) menu.
- While focus is present on [IOR panel](#), choose [\[Save IOR Data As...\]](#) from the [\[File\]](#) menu.
- While focus is present on [Call Stack panel](#), choose [\[Save Call Stack Data As...\]](#) from the [\[File\]](#) menu.
- While focus is present on [Local Variables panel](#), choose [\[Save Local Variable Data As...\]](#) from the [\[File\]](#) menu.
- While focus is present on [Output panel](#), choose [\[Save Tab name As...\]](#) from the [\[File\]](#) menu.

[Description of each area]

(1) [Save in] area

Select the folder in which you want to save the file.

(2) File list area

Displays a list of files that match the conditions selected in the [\[Save In\]](#) area and [\[File Type\]](#) area.

(3) [File name] area

Specify a file name under which you want to save.

(4) [Save as type] area**(a) For the Editor panel**

Depending on the type of file being edited, this area displays the following file types.

Text file (*.txt)	Text format
C source file (*.c)	C language source file
C++ source file (*.cpp;*.cp;*.cc)	C++ source file
Header file (*.h;*.hpp;*.inc)	Header file
Assemble file (*.src)	Assembler source file
Link order specification file (*.mtls)	Link order specification file
Map file (*.map;*.lbp)	Map file
Hex file (*.hex)	Hex file
Assemble list file (*.lst)	Assembler list file
S record file (*.mot)	S record file

(b) For the CPU Register panel, Watch panel, IOR panel, Call Stack panel, and Local Variables panel

Following file types are displayed.

The panel content is saved to a file in the file format selected from the drop-down list.

Text file (*.txt)	Text format (default)
CSV (Comma-Separated) (*.csv)	CSV format ^{Note}

Note Each piece of data are separated with a comma (,) when saved.

To avoid the problem of an invalid file format in cases where any data includes a comma (,), each piece of data are enclosed in double-quotes (" ") when they are output to a file.

(c) For the Output panel

Following file types are displayed.

Data can be saved in only text format.

Text file (*.txt)	Text format (default)
-------------------	-----------------------

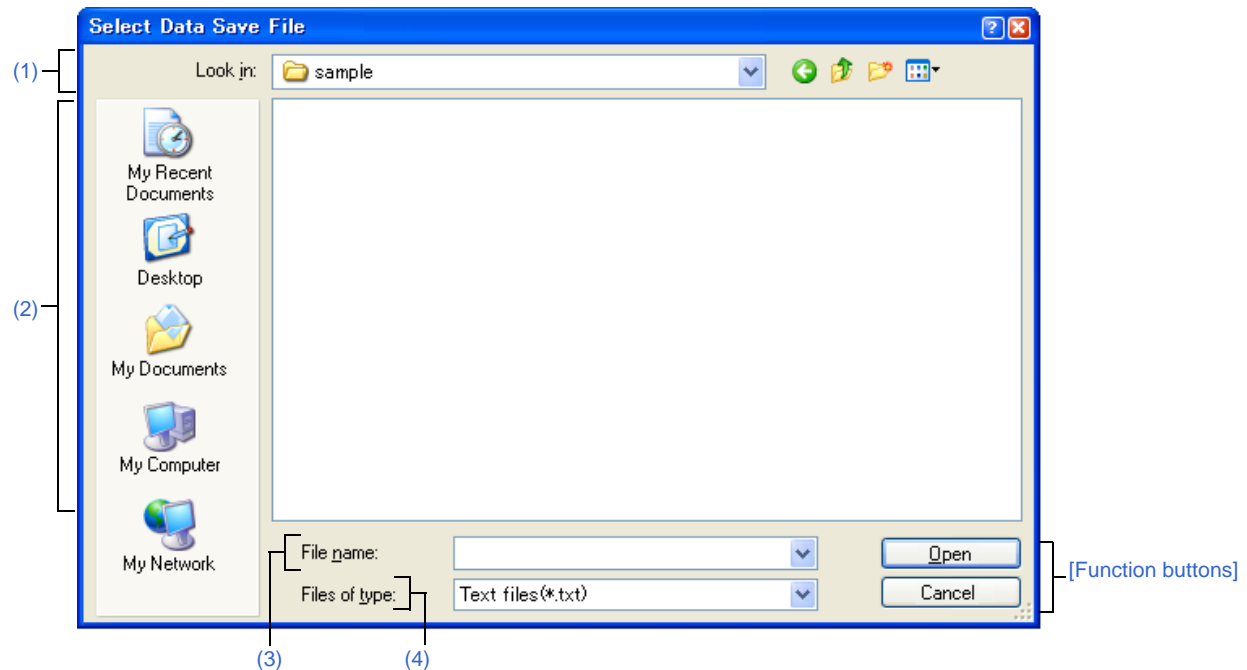
[Function buttons]

Button	Function
Save	Saves a file with a specified file name.
Cancel	Closes this dialog box.

Select Data Save File dialog box

Select a file in which to save the data.

Figure A-78. Select Data Save File Dialog Box



This section describes the following.

- [\[How to open\]](#)
- [\[Description of each area\]](#)
- [\[Function buttons\]](#)

[How to open]

- Click the [...] button in the [File Name] area of the [Data Save dialog box](#).

[Description of each area]

(1) [Look in] area

From the drop-down list, select the folder which contains the file you wish to save.

(2) File list area

Displays a list of files that match the conditions selected in the [Look in] area and [Files of type] area.

(3) [File name] area

Specify the name of a file you wish to save.

(4) [Files of type] area

Select the type of file to save (file type) from the drop-down list below.

The selectable file format is determined by the type of data you are saving.

(a) When saving the displayed content of a panel

Text files (*.txt)	Text format (default)
CSV (Comma-Separated Variables)(*.csv)	CSV format ^{Note}

Note The data is saved with entries separated by commas (,).

If the data contains commas, each entry is surrounded by double quotes (" ") in order to avoid illegal formatting.

(b) When saving upload data

For the selectable file format, see "[Table 2-3. Uploadable File Formats](#)".

[Function buttons]

Button	Function
Open	Specifies a specified file in the Data Save dialog box .
Cancel	Closes this dialog box.

APPENDIX B INPUT/OUTPUT FUNCTIONS

This section describes the input/output settings for the simulator in a low-level interface routine (assembly language part).

The simulator provides functions for handling standard I/O and file I/O.

B.1 Standard Input/Output and File Input/Output

The content of a file that is called from a low-level interfaced routine to perform input and output is replaced with a program for input/output.

With the standard input/output functions [GETC](#) and [PUTC](#), the functions that perform 1-character input and output, "charput," "charget" (_charput, _charget), are replaced with a program for debug console facilities, to perform input and output of data onto and from the [Debug Console panel](#).

The input/output functions are listed below.

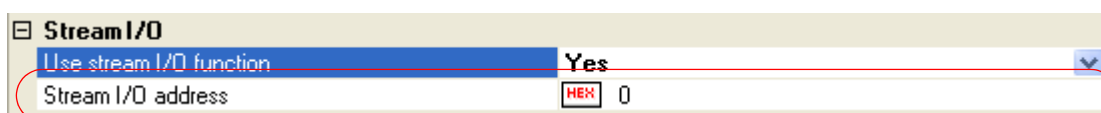
Table B-1. Input/Output Facilities

Classification	Facility name	Description
Standard input/output	GETC	Accepts 1 byte from standard input.
	PUTC	Outputs 1 byte to standard output.
File input/output	FOPEN	Opens a file.
	FCLOSE	Closes a file.
	FGETC	Accepts 1 byte from a file.
	FPUTC	Outputs 1 byte to a file.
	FEOF	Checks termination of a file.
	FSEEK	Moves the file pointer to a specified position.
	FTELL	Obtains the file pointer's current position.

Remark For details about low-level interface routines, see Chapter 7, "Startup," in the separate edition, "CubeSuite+ RX Coding."

To implement I/O function of the simulator, first specify the I/O address in the [\[Stream I/O address\]](#) property within the [\[Stream I/O\] \[Simulator\]](#) category on the [Property panel's \[Debug Tool Settings\] tab](#). This can be done by entering address expressions in the range of 0 to "last address of address space" directly from the keyboard.

Figure B-1. [Stream I/O] Category



Upon detecting "Branch to subroutine" instruction (BSR, JSR) to a specified address while executing instructions of the program, the simulator performs an input/output process using the contents of the R1 and R2 registers as parameters.

Therefore, be sure that the following setting is made in the program before an input/output process is performed.

- Function code (R1 register)

MSB	1 byte	1 byte	1 byte	1 byte	LSB
	0x01	Function code	0x00	0x00	

- Address of the parameter block (R2 register)

For the content of the parameter block, see the description of each input/output function.

MSB		LSB

- Reservation of areas for the parameter block and stream input/output buffer

To access each parameter in the parameter block, make sure that an access is made in size of the relevant parameter.

GETC

Accepts 1 byte from standard input.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x21	0x00	0x00

[Parameter Block (R2 Register)]

	1 byte	1 byte
+0	Beginning address of input buffer	
+2		

[Parameters]

Parameter	Description
Beginning address of input buffer (input)	Beginning address of the input buffer to which input data is written

[Detailed Description]

- Accepts 1 byte from standard input.

[Example]

```

_charget:
    MOV.L    #1210000h,R1    ; Set the function code of GETC in R1.
    MOV.L    #IO_BUF,R2     ; Set the address of the input buffer in R2.
    MOV.L    #PARM,R3       ; Set the address of the parameter block in R3.
    MOV.L    R2,[R3]        ; Set the address of the input buffer in the parameter
                                ; block.
    MOV.L    R3,R2          ; Set the value of R3 (address of the parameter block)
                                ; in R2.
    MOV.L    #SIM_IO,R3     ; Set the address of the system call in R3.
    JSR      R3              ; System call
    MOV.L    #IO_BUF,R2     ; Set the address of the input buffer in R2.
    MOVU.B   [R2],R1        ; Set the first byte of the input buffer
                                ; (acquired 1-byte character) in R1.
    RTS                               ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL   1        ; Parameter block area
    .SECTION  B_1,DATA
IO_BUF:    .BLKL   1        ; Input/output buffer

```


PUTC

Outputs 1 byte to standard output.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x22	0x00	0x00

[Parameter Block (R2 Register)]

1 byte	1 byte
+0	<i>Beginning address of</i>
+2	<i>output buffer</i>

[Parameters]

Parameter	Description
<i>Beginning address of output buffer</i> (input)	Beginning address of the output buffer that contains output data

[Detailed Description]

- Outputs 1 byte to standard output.

[Example]

```

_charput:
    MOV.L    #IO_BUF,R2    ; Set the address of the output buffer in R2.
    MOV.B    R1,[R2]       ; Set the value of R1 (output character) in the output
                           ; buffer.
    MOV.L    #1220000h,R1   ; Set the function code of PUTC in R1.
    MOV.L    #PARM,R3      ; Set the address of the parameter block in R3.
    MOV.L    R2,[R3]       ; Set the address of the output buffer in the output
                           ; buffer.
    MOV.L    R3,R2         ; Set the value of R3 (address of the parameter block)
                           ; in R2.
    MOV.L    #SIM_IO,R3    ; Set the address of the system call in R3.
    JSR      R3            ; System call
    RTS                      ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL    1      ; Parameter block area
    .SECTION  B_1,DATA
IO_BUF:    .BLKL    1      ; Input/output buffer

```

FOPEN

Opens a file.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x25	0x00	0x00

[Parameter Block (R2 Register)]

	1 byte	1 byte
+0	<i>Execution result</i>	<i>File number</i>
+2	<i>Open mode</i>	<i>Unused</i>
+4	<i>Beginning address of</i>	
+6	<i>file name</i>	

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Terminated normally -1: Error
<i>File number</i> (output)	Number that is used for access to files after the open process
<i>Open mode</i> (input)	0x00: "r" 0x01: "w" 0x02: "a" 0x03: "r+" 0x04: "w+" 0x05: "a+" 0x10: "rb" 0x11: "wb" 0x12: "ab" 0x13: "r+b" 0x14: "w+b" 0x15: "a+b" The content of each mode is as follows: "r": Opens for read. "w": Opens a blank file for write. "a": Opens for write from the end of a file. "r+": Opens for read and write. "w+": Reads a blank file and opens it for write. "a+": Opens additionally for read. "b": Opens in binary mode.
<i>Beginning address of file name</i> (input)	Beginning address of the area that contains the file name

[Detailed Description]

- When a file is opened by [FOPEN], a file number is returned.
This file number is used in subsequent operations to input or output to the file, as well as to close the file.
- Up to 256 files can be opened at a time.

[Example]

```

_fileopen:
    MOV.L    R2,R5        ; Set the value of R2 (open mode) in R5.
    MOV.L    #PARM,R2     ; Set the address of the parameter block in R2.
    MOV.L    R1,4h:5[R2]  ; Set the value of R1 (first address of the filename)
                        ; in R2 + 4 bytes.
    MOV.B    R5,2h:5[R2]  ; Set the value of R5 in R2 + 2 bytes (open mode).
    MOV.L    #01250000h,R1 ; Set the function code of FOPEN in R1.
    MOV.L    #SIM_IO,R5   ; Set the address of the system call in R5.
    JSR      R5           ; System call
    NOP
    MOV.L    #PARM,R2     ; Set the address of the parameter block in R3.
    MOV.B    1h:5[R2],R1  ; Set the value of R2 + 1 byte (file number) in R1.
    MOV.B    R1,[R3]      ; Set the value of R1 in the location pointed to by
                        ; R3 (file number pointer).
    MOV.B    [R2],R1      ; Set the first byte of R2 (result of execution) in R1.
    RTS                ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL  2        ; Parameter block area

```

FCLOSE

Closes a file.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x06	0x00	0x00

[Parameter Block (R2 Register)]

1 byte	1 byte
+0	<i>Execution result</i> <i>File number</i>

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Terminated normally -1: Error
<i>File number</i> (input)	Number that is returned when a file is opened

[Detailed Description]

Closes a file.

[Example]

```

_fileclose:
    MOV.L    #PARM,R2        ; Set the address of the parameter block in R2.
    MOV.B    R1,1h:5[R2]    ; Set the value of R1 (file number) in R2 + 1 bytes.
    MOV.L    #01060000h,R1  ; Set the function code of FCLOSE in R1.
    MOV.L    #SIM_IO,R3     ; Set the address of the system call in R3.
    JSR      R3              ; System call
    NOP
    MOV.L    #PARM,R2        ; Set the address of the parameter block in R2.
    MOV.B    [R2],R1         ; Set the first byte of R2 (result of execution) in R1.
    RTS
    ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL    1        ; Parameter block area

```

FGETC

Reads 1 byte of data from a file.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x27	0x00	0x00

[Parameter Block (R2 Register)]

	1 byte	1 byte
+0	<i>Execution result</i>	<i>File number</i>
+2	Unused	
+4	<i>Beginning address of</i>	
+6	<i>input buffer</i>	

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Terminated normally -1: EOF detected
<i>File number</i> (input)	Number that is returned when a file is opened
<i>Beginning address of input buffer</i> (input)	Beginning address of the buffer to which input data is written

[Detailed Description]

- Reads 1 byte of data from a file.

[Example]

```

_fcharget:
    MOV.L    R2,R5          ; Set the value of R2 (file number) in R5.
    MOV.L    #PARM,R2       ; Set the address of the parameter block in R2.
    MOV.L    R1,4h:5[R2]    ; Set the value of R1 (input buffer) in R2 + 4 bytes.
    MOV.B    R5,1h:5[R2]    ; Set the value of R5 in R2 + 1 bytes (file number).
    MOV.L    #01270000h,R1  ; Set the function code of FGETC in R1.
    MOV.L    #SIM_IO,R3     ; Set the address of the system call in R3.
    JSR      R3             ; System call
    NOP
    MOV.L    #PARM,R1       ; Set the address of the parameter block in R1.
    MOV.B    [R1],R1        ; Set the first byte of R1 (result of execution) in R1.
    RTS
    ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL    2          ; Parameter block area

```

FPUTC

Writes 1 byte of data to a file.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x28	0x00	0x00

[Parameter Block (R2 Register)]

	1 byte	1 byte
+0	<i>Execution result</i>	<i>File number</i>
+2	Unused	
+4	<i>Beginning address of</i>	
+6	<i>output buffer</i>	

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Terminated normally -1: Error
<i>File number</i> (input)	Number that is returned when a file is opened
<i>Beginning address of output buffer</i> (input)	Beginning address of the buffer that contains output data

[Detailed Description]

- Writes 1 byte of data to a file.

[Example]

```

_fcharput:
    MOV.L    R2,R5        ; Set the value of R2 (file number) in R5.
    MOV.L    #PARM,R2     ; Set the address of the parameter block in R2.
    MOV.L    #IO_BUF,R4   ; Set the address of the output buffer in R4.
    MOV.L    R4,4h:5[R2]  ; Set the value of R4 (output buffer) in R2 + 4 bytes.
    MOV.B    R1,[R4]      ; Set the value of R1 (output character) in the
                          ; location pointed to by R4 (output buffer).
    MOV.B    R5,1h:5[R2]  ; Set the value of R5 in R2 + 1 bytes (file number).
    MOV.L    #01280000h,R1 ; Set the function code of FPUTC in R1.
    MOV.L    #SIM_IO,R3   ; Set the address of the system call in R3.
    JSR      R3           ; System call
    NOP
    MOV.L    #PARM,R1     ; Set the address of the parameter block in R1.
    MOV.B    [R1],R1      ; Set the first byte of R1 (result of execution) in R1.
    RTS                          ; Return to the address where the function was called.

    .SECTION B,DATA,ALIGN=4
PARM:      .BLKL    2      ; Parameter block area
    .SECTION B_1,DATA
IO_BUF:    .BLKL    1      ; Input/output buffer

```

FEOF

Checks for end of file.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x0B	0x00	0x00

[Parameter Block (R2 Register)]

1 byte	1 byte		
+0	<table border="1"> <tr> <td><i>Execution result</i></td> <td><i>File number</i></td> </tr> </table>	<i>Execution result</i>	<i>File number</i>
<i>Execution result</i>	<i>File number</i>		

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Not EOF -1: EOF detected
<i>File number</i> (input)	Number that is returned when a file is opened

[Detailed Description]

- Checks for end of file.

[Example]

```

_ffeof:
    MOV.L    #PARM,R2        ; Set the address of the parameter block in R2.
    MOV.B    R1,1h:5[R2]     ; Set the value of R1 (file number) in R2 + 1 bytes.
    MOV.L    #010B0000h,R1   ; Set the function code of FEOF in R1.
    MOV.L    #SIM_IO,R3      ; Set the address of the system call in R3.
    JSR      R3              ; System call
    NOP
    MOV.L    #PARM,R2        ; Set the address of the parameter block in R2.
    MOV.B    [R2],R1         ; Set the first byte of R2 (result of execution) in R1.
    RTS
    ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL    1        ; Parameter block area

```


FSEEK

Moves the file pointer to a specified position.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x0C	0x00	0x00

[Parameter Block (R2 Register)]

	1 byte	1 byte
+0	<i>Execution result</i>	<i>File number</i>
+2	<i>Direction</i>	Unused
+4	<i>Offset</i>	
+6		

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Terminated normally -1: Error
<i>File number</i> (input)	Number that is returned when a file is opened
<i>Direction</i> (input)	0: Offset in bytes from the beginning of a file 1: Offset in bytes from the current file pointer 2: Offset in bytes from the tail end of a file
<i>Offset</i> (input)	Number of bytes from the position specified by direction

[Detailed Description]

- Moves the file pointer to a specified position.

[Example]

```
_fpseek:
    MOV.L    R2,R5          ; Set the value of R2 (offset) in R5.
    MOV.L    #PARM,R2       ; Set the address of the parameter block in R2.
    MOV.L    R5,4h:5[R2]    ; Set the value of R5 in R1 + 4 bytes (offset).
    MOV.B    R1,1h:5[R2]    ; Set the value of R1 (file number) in R1 + 1 bytes
                           ; (offset).
    MOV.B    R3,2h:5[R2]    ; Set the value of R3 (direction) in R2 + 2 bytes
                           ; (direction).
    MOV.L    #010C0000h,R1   ; Set the function code of FSEEK in R1.
    MOV.L    #SIM_IO,R5     ; Set the address of the system call in R5.
    JSR      R5             ; System call
    NOP
    MOV.L    #PARM,R1       ; Set the address of the parameter block in R1.
    MOV.B    [R1],R1        ; Set the first byte of R1 (result of execution) in R1.
    RTS                               ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:      .BLKL    2          ; Parameter block area
```

FTELL

Obtains the current position of the file pointer.

[Function Code (R1 Register)]

1 byte	1 byte	1 byte	1 byte
0x01	0x0D	0x00	0x00

[Parameter Block (R2 Register)]

	1 byte	1 byte
+0	<i>Execution result</i>	<i>File number</i>
+2	Unused	
+4	<i>Offset</i>	
+6		

[Parameters]

Parameter	Description
<i>Execution result</i> (output)	0: Terminated normally -1: Error
<i>File number</i> (input)	Number that is returned when a file is opened
<i>Offset</i> (output)	Current position of the file pointer (Number of bytes from the top of the file)

[Detailed Description]

- Obtains the current position of the file pointer.

[Example]

```
_ftell:
    MOV.L    R2,R5        ; Set the value of R2 (offset) in R5.
    MOV.L    #PARM,R2     ; Set the address of the parameter block in R2.
    MOV.B    R1,1h:5[R2]  ; Set the value of R1 (file number) in R2 + 1 bytes
                          ; (direction).
    MOV.L    #010D0000h,R1 ; Set the function code of FTELL in R1.
    MOV.L    #SIM_IO,R3   ; Set the address of the system call in R3.
    JSR      R3           ; System call
    NOP
    MOV.L    #PARM,R2     ; Set the address of the parameter block in R2.
    MOV.L    4h:5[R2],R1  ; Set the value of R2 + 4 bytes (new offset) in R1.
    MOV.L    R1,[R5]      ; Set the value of R1 in the location pointed to by R5
                          ; (offset pointer).
    MOV.B    [R2],R1      ; Set the first byte of R2 (result of execution) in R1.
    RTS                      ; Return to the address where the function was called.

    .SECTION  B,DATA,ALIGN=4
PARM:    .BLKL    2        ; Parameter block area
```

APPENDIX C INDEX

A

Access width ... 321

Access-related event ... 127, 159

Action event ... 151, 310, 328

Action Events dialog box ... 328

 [Interrupt Event Setting] tab ... 333

 [Printf event] tab ... 330

Address range ... 321

After-execution break ... 85

AND ... 93, 163

Array ... 275, 281

Auto variables ... 275

B

Before-execution break ... 83

Big endian ... 104

Binary data format ... 50, 53, 56

Break ... 23, 34

Break event ... 85, 92, 168

Break event (access-related) ... 87, 166, 167

Break event (execution-related) ... 85, 166, 167

Break factor ... 83, 93

Breakpoint ... 23, 83

Built-in event ... 125, 142, 157, 307

C

Call Stack panel ... 293

Call-stack information ... 293

Channel number ... 144, 308

Clock ... 17, 28, 39

Code ... 254

Code coverage measurement ... 148

Code coverage measurement result ... 235

Code outlining ... 234

Collecting an execution history ... 121

 Clearing the trace memory ... 133

 Collecting an execution history in a section ... 125

 Collecting an execution history only when conditions are met ... 130

 Collecting an execution history up to a halt ... 125

 Displaying an execution history ... 131

 Saving the displayed content of an execution history ... 139

 Searching for trace data ... 134

 Setting up a trace operation ... 121

Combination break ... 92, 162, 166, 167, 306, 307

Combination condition ... 92, 129, 162, 353

Combination Condition dialog box ... 352

Configuration of operating environment ... 14

Connect to/disconnect from the debug tool ... 44

Connectable debug tool ... 12

Connection to the debug tool ... 44

Control register ... 106, 262

Coverage measurement ... 148

 Configure the coverage measurement ... 148

 Display coverage measurement results ... 148

CPU Register panel ... 262

Current PC mark ... 253

Current PC position ... 232, 253

D

Data coverage measurement ... 148

Data flash memory ... 17, 28

Data Save dialog box ... 380

Debug Console panel ... 318

Debug information ... 224

Debug toolbar ... 188

Debug-only project ... 47

Detailed Settings of Access Events dialog box ... 362

Detailed Settings of Execution Events dialog box ... 358

Detailed Settings of Interrupt Events dialog box ... 372

Detailed Settings of Timer Measurement dialog box ... 146, 355

Disassemble display mode ... 132, 300

Disassemble panel ... 251

Disassembled text ... 66, 251

Display format of watch-expression ... 285

Display function call information from the stack ... 119

- Display call stack information ... 119
- Saving the displayed contents of call-stack information ... 120

Display the result of disassembling. ... 66

Displaying and changing memory, registers, and variables ... 95

- Displaying and changing global and static variables ... 111
- Displaying and changing local variables ... 111
- Displaying and changing memory contents ... 95
- Displaying and changing the CPU registers ... 106
- Displaying and changing the I/O registers ... 108
- Displaying and changing watch expressions ... 113

Displaying and changing programs ... 60

- Displaying source files ... 60
- Displaying the result of disassembling ... 66
- Executing a build in parallel with other processes ... 69
- Performing line assembly ... 70

DMM (Dynamic Memory Modification) function ... 98

Download ... 25, 37, 47, 223

Download condition ... 52, 323

Download Files dialog box ... 323

Downloadable File Formats ... 50

E

Editor panel ... 228

Encoding ... 336

Encoding dialog box ... 335

Endian ... 20, 22, 33, 39, 103

Epilog of functions ... 275

Event area ... 232, 253

Event management ... 156

- Changing states of event setting ... 157
- Deleting events ... 165
- Displaying only a specific type of event ... 158
- Editing detailed settings of events ... 158
- Jump to an event address ... 158

- Limitation on the number of valid events ... 166
- Types of events that can be set or removed during execution ... 168

Event mark ... 84, 126, 144, 306

Event type ... 307

Events panel ... 304

Execute programs ... 78

- Execute a specified routine ... 81
- Execute programs ... 78
- Execute programs in steps ... 80

Execution only once ... 146, 356

Execution-related event ... 126, 158

External flash memory ... 20, 57

External flash memory dialog box ... 392

F

Features ... 8

File input/output ... 406

File monitor ... 235

G

[General - Build/Debug] category ... 390

[General - Font and Color] category ... 385

General-purpose registers ... 106, 262

Global variable ... 111

Go to Here ... 79, 167

Go to Line dialog box ... 378

Go to the Location dialog box ... 379

H

Hardware break ... 83, 166, 167, 168, 306, 307

Hex format ... 50, 53, 56

Hook transaction ... 226

Hot plug-in connection ... 45

Hot-plug Adapter ... 45

I

I/O registers protected against read ... 270

ID code ... 19

Initialization data ... 103, 338

Input format of watch-expression ... 282

Input/output functions ... 406

FCLOSE ... 412
FEOF ... 416
FGETC ... 413
FOPEN ... 410
FPUTC ... 414
FSEEK ... 417
FTELL ... 419
GETC ... 408
PUTC ... 409
Instruction level debugging ... 60
Internal static variables ... 275
Interrupt event ... 153, 167, 168, 306, 307, 372
IOR panel ... 268

J

Jump to subroutine instruction ... 81

L

Label ... 300
Label line ... 254
Line assembly ... 70, 255
Little endian ... 104
Load module file ... 47
Load module format ... 50
Local variable ... 100, 111, 168
Local Variables panel ... 274
Lock bit ... 58
Lost ... 100
Low-level interface routine ... 172

M

Main area ... 232
Main window ... 185
Maximum number of enabled events ... 166
Measure Execution Time of the Program ... 141
Measuring the execution time ... 141
 Measuring execution time from start to stop ... 141
 Measuring execution time in a section ... 143
 Range of measurable time ... 147
 Setting the timer measurement operation ... 141
Memory access ... 22, 33, 41
Memory Initialize dialog box ... 338

Memory mapping ... 21, 32, 40, 320
Memory Mapping dialog box ... 320
Memory panel ... 244
Memory Search dialog box ... 340
Memory settings ... 40
Memory type ... 320
Menubar ... 186
Methods to register watch expressions ... 114
Mixed display mode ... 132, 300
Motorola S format ... 53, 56
Motorola S type Hex format ... 50
Moving to a specified address ... 67
Moving to a specified line ... 62
Moving to a symbol definition part ... 68

N

Newline code ... 336
Number of trace frames ... 125

O

Offset value ... 254, 300
Open File dialog box ... 398
Open Log File dialog box ... 394
Operation mode ... 58
Option dialog box ... 383
 [General - Build/Debug] category ... 390
 [General - Font and Color] category ... 385
OR ... 93, 163
Output panel ... 315
Overlay section ... 49, 75

P

Pass count ... 159, 360
Point trace ... 167, 168, 306, 307
Point trace event ... 130, 242
Pointer-type variables ... 275, 281
Pop-up display of variables ... 235
Port Setting dialog box ... 375
Power-down mode ... 168
Print Address Range Settings dialog box ... 342
Print Preview window ... 400
Printf ... 306, 307

Printf event ... 151, 167, 168, 330

Priority section ... 75, 76

Progress Status dialog box ... 382

Project Tree panel ... 193

Prolog of functions ... 275

Property panel ... 196

 [Connect Settings] tab ... 199

 [Debug Tool Settings] tab ... 209

 [Download File Settings] tab ... 223

 [Hook Transaction Settings] tab ... 226

Pseudo-RRM ... 98

R

Rapid build function ... 69

Realtime display update function ... 98, 118

Real-time RAM monitor ... 122

Realtime RAM monitor function ... 98

Register Action Event ... 151, 153

Register variables ... 275

Registering a watch-expression ... 279

Reset ... 98, 141

Reset event ... 164, 354

Reset microcontroller (CPU) ... 78

Return out execution ... 81, 189

RRM (Real-time RAM Monitor) ... 98

RRM function ... 167

Run-Break time ... 141

Run-Break Timer ... 306, 307

Run-Break Timer event ... 142, 305

S

Save As dialog box ... 402

Save Settings dialog box ... 336

Scope selection ... 111

Scroll Range Settings dialog box ... 376

Select Data Save File dialog box ... 404

Select Download File dialog box ... 396

Selecting blocks ... 236

Sequential ... 93, 163

Set an action into programs ... 151

 Insert an interrupt event ... 153

 Insert printf ... 151

Set PC to Here ... 80

Set states of events ... 157

Setting up a trace operation ... 121

Setting up the hook process ... 169

Software break ... 83, 166, 167, 168, 306, 307

Source display mode ... 132, 300

Source level debugging ... 60, 228

Specified routine ... 81

Specify the PIC/PID offset ... 73

Standard input/output ... 172, 406

Standard library function ... 172

Statusbar ... 190

Step execution ... 80

Step in execution ... 80, 189

Step over execution ... 81, 189

Stop programs ... 83

 Stop the program at the arbitrary position (break event) ... 85

 Stop the program at the arbitrary position (breakpoint) ... 83

 Stop the program manually ... 83

 Stop the program with the access to variables ... 87

Structure ... 275, 281

Subjects that can be registered as watch expressions ... 114

Supervisor mode ... 82, 146

T

Tag jump ... 64, 234, 316

Target range for the RRM function ... 99

Targets that can be registered as watch expressions ... 281

Text Edit dialog box ... 327

Timer end event ... 143, 307

Timer measurement ... 141, 166, 167, 306, 307, 356

Timer measurement event ... 143

Timer start event ... 143, 307

Trace ... 122, 167, 168, 306, 309

Trace end ... 307

Trace end event ... 126

Trace event ... 125
Trace memory ... 121, 124
Trace number ... 299
Trace panel ... 297
Trace Search dialog box ... 344
Trace start ... 307
Trace start event ... 126
Type of breakpoint ... 83, 216
Types of events ... 168

U

Unconditional trace ... 306, 307
Unconditional trace event ... 125, 157
Union ... 275, 281
Upload ... 47, 58
Uploadable file formats ... 59
Usage of PIC/PID Function ... 72
USD file ... 57

V

Verify ... 215

W

Watch panel ... 279
Watchdog timer ... 98
Watch-expression ... 113, 279
Window reference ... 183
Work RAM ... 20

Z

Zoom in or out on a view ... 236

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Oct 01, 2011	-	First Edition issued

CubeSuite+ V1.01.00
User's Manual: RX Debug

Publication Date: Rev.1.00 Oct 01, 2011

Published by: Renesas Electronics Corporation



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.

13F, 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

CubeSuite+ V1.01.00