User's Manual

CS+ V8.06.00

Integrated Development Environment

User's Manual: RX Debug Tool

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).

Renesas Electronics
www.renesas.com

Rev.1.00 2021.06

Notice

1. 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

© 2021 Renesas Electronics Corporation. All rights reserved.

How to Use This Manual

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

CS+ 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 CS+ and design software and hardware application systems.
Purpose	This manual is intended to give users an understanding of the functions of the CS+ 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. 1.GENERAL 2.FUNCTIONS A.WINDOW REFERENCE B.I/O FUNCTIONS
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:	XXX (overscore over pin or signal name)
Note:	Footnote for item marked with Note in the text
Caution:	Information requiring particular attention
Remarks:	Supplementary information
Numeric representation:	Decimal ... XXXX Hexadecimal ... 0xXXXX

TABLE OF CONTENTS

1. GENERAL 10
1.1 Summary 10
1.2 Features. 10
2. FUNCTIONS. 11
2.1 Overview 11
2.2 Preparation before Debugging. 13
2.2.1 Confirm the connection to a host machine 13
2.2.1.1 [E1] 13
2.2.1.2 [E20] 14
2.2.1.3 [EZ Emulator]. 14
2.2.1.4 [Simulator] 15
2.3 Configuration of Operating Environment of the Debug Tool 16
2.3.1 Select the debug tool to use 16
2.3.2 [E1] 17
2.3.2.1 [Connect Settings] tab 18
2.3.2.2 [Debug Tool Settings] tab 23
2.3.2.3 [Download File Settings] tab 30
2.3.2.4 [Hook Transaction Settings] tab. 30
2.3.3 [E20] 31
2.3.3.1 [Connect Settings] tab 32
2.3.3.2 [Debug Tool Settings] tab 36
2.3.3.3 [Download File Settings] tab 44
2.3.3.4 [Hook Transaction Settings] tab. 44
2.3.4 [EZ Emulator] 45
2.3.4.1 [Connect Settings] tab 45
2.3.4.2 [Debug Tool Settings] tab 48
2.3.4.3 [Download File Settings] tab 54
2.3.4.4 [Hook Transaction Settings] tab. 54
2.3.5 [Simulator]. 55
2.3.5.1 [Connect Settings] tab 55
2.3.5.2 [Debug Tool Settings] tab 56
2.3.5.3 [Download File Settings] tab 60
2.3.5.4 [Hook Transaction Settings] tab. 60
2.4 Connect to/Disconnect from the Debug Tool 61
2.4.1 Connect the debug tool to CS+ 61
2.4.2 Disconnect the debug tool from CS+ 63
2.4.3 Connect the debug tool to CS+ using hot plug-in [E1(JTAG)] [E20(JTAG)] 63

2.5 Download and Upload . . . 66
2.5.1 Execute downloading . . . 66
2.5.2 An applied method of download . . . 68
2.5.2.1 Changing the download conditions for load module files . . . 69
2.5.2.2 Adding a download file (*.hex, *.mot, or *.bin) . . . 71
2.5.2.3 Downloading multiple load module files . . . 72
2.5.2.4 Performing source-level debug with hex format, Motorola S format, or binary data format files . . . 74
2.5.2.5 Downloading files to external flash memory [E1] [E20] . . . 75
2.5.3 Executing an upload . . . 77
2.6 Displaying and Changing Programs . . . 79
2.6.1 Displaying source files . . . 79
2.6.2 Displaying the disassembled result . . . 80
2.6.2.1 Changing the display mode . . . 80
2.6.2.2 Changing the display form . . . 81
2.6.2.3 Moving to a specified address . . . 82
2.6.2.4 Moving to a symbol definition part . . . 82
2.6.2.5 Saving the displayed contents of disassembled results . . . 82
2.6.3 Executing a build in parallel with other processes . . . 83
2.6.4 Performing line assembly . . . 84
2.6.4.1 Editing instructions . . . 84
2.6.4.2 Editing instruction code . . . 85
2.7 Usage of PIC/PID Function . . . 86
2.7.1 Changing the allocation of a load module using the PIC/PID function . . . 86
2.8 Setting Overlay Sections . . . 89
2.8.1 Selecting the priority section . . . 89
2.9 Execute Programs . . . 92
2.9.1 Reset microcontroller (CPU) . . . 92
2.9.2 Execute programs . . . 92
2.9.2.1 Execute after resetting microcontroller (CPU) . . . 93
2.9.2.2 Execute from the current address . . . 93
2.9.2.3 Execute after changing PC value . . . 94
2.9.3 Execute programs in steps . . . 94
2.9.3.1 Step into the function (Step In execution) . . . 95
2.9.3.2 Step over the function (Step Over execution) . . . 95
2.9.3.3 Execute until return is completed (Return Out execution) . . . 95
2.9.4 Execute a specified routine [E1] [E20] [EZ Emulator] . . . 96
2.10 Stop Programs (Break) . . . 99
2.10.1 Stop the program manually . . . 99
2.10.2 Stop the program at the arbitrary position (breakpoint) . . . 99
2.10.2.1 Set the type of breakpoints/break timing to use [E1] [E20] [EZ Emulator] . . . 99
2.10.2.2 Set a breakpoint . . . 99
2.10.2.3 Edit a hardware breakpoint . . . 100

2.10.2.4 Delete a breakpoint . . . 101
2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20] [EZ Emulator]. . . . 101
2.10.3.1 Set a beak event (execution-related). . . . 101
2.10.3.2 Edit a break event (execution-related). . . . 102
2.10.3.3 Delete a break event (execution-related) . . . 102
2.10.4 Stop the program with the access to variables/I/O registers. . . . 102
2.10.4.1 Set a break event (access-related) to a variable/I/O register . . . 103
2.10.4.2 Edit a break event (access-related) to a variable/IO register . . . 107
2.10.4.3 Delete a break event (access-related) to a variable/IO register . . . 107
2.10.5 Set multiple break events in combination (Combination break) [E1] [E20] [EZ Emulator] . . . 107
2.10.6 Other break factors . . . 108
2.11 Displaying and Changing Memory, Registers, and Variables . . . 110
2.11.1 Displaying and changing memory contents . . . 110
2.11.1.1 Specifying the display position. . . . 110
2.11.1.2 Changing the display form of values . . . 111
2.11.1.3 Changing memory contents. . . . 112
2.11.1.4 Displaying and changing memory contents during program execution . . . 113
2.11.1.5 Searching for memory contents. . . . 116
2.11.1.6 Collectively changing (initializing) memory contents . . . 117
2.11.1.7 Saving displayed memory contents. . . . 119
2.11.2 Displaying and changing the CPU registers . . . 122
2.11.2.1 Changing the form in which values are displayed . . . 122
2.11.2.2 Changing the CPU register contents . . . 123
2.11.2.3 Saving the displayed CPU register contents . . . 123
2.11.3 Displaying and changing the I/O registers . . . 123
2.11.3.1 Searching I/O register . . . 124
2.11.3.2 Putting the I/O registers in order . . . 124
2.11.3.3 Changing the form in which values are displayed . . . 125
2.11.3.4 Changing the contents of I/O registers . . . 125
2.11.3.5 Displaying and changing I/O register contents during program execution . . . 125
2.11.3.6 Saving the displayed I/O register contents . . . 126
2.11.4 Displaying and changing global and static variables . . . 126
2.11.5 Displaying and changing local variables. . . . 126
2.11.5.1 Changing the form in which values are displayed . . . 127
2.11.5.2 Changing the contents of local variables. . . . 127
2.11.5.3 Saving the displayed contents of local variables . . . 128
2.11.6 Displaying and changing watch-expressions . . . 128
2.11.6.1 Registering watch-expressions . . . 129
2.11.6.2 Putting the registered watch-expressions in order . . . 130
2.11.6.3 Editing the registered watch-expressions . . . 131
2.11.6.4 Removing watch-expressions . . . 131
2.11.6.5 Changing the form in which values are displayed . . . 131

2.11.6.6 Changing the contents of watch-expressions . . . 132
2.11.6.7 Displaying and changing the contents of watch-expressions during program execution . . . 132
2.11.6.8 Exporting/importing watch-expressions . . . 132
2.11.6.9 Saving the displayed contents of watch-expressions . . . 134
2.12 Display Function Call Information from the Stack . . . 135
2.12.1 Display call stack information . . . 135
2.12.1.1 Changing the form in which values are displayed . . . 135
2.12.1.2 Jumping to the source line . . . 136
2.12.1.3 Displaying local variables . . . 136
2.12.1.4 Saving the displayed contents of call stack information . . . 136
2.13 Collecting an Execution History . . . 137
2.13.1 Setting up a trace operation . . . 137
2.13.1.1 For [E1] . . . 137
2.13.1.2 For [E20] . . . 139
2.13.1.3 For [EZ Emulator] . . . 142
2.13.1.4 For [Simulator] . . . 143
2.13.2 Collecting an execution history up to a halt . . . 144
2.13.3 Collecting an execution history in a section . . . 144
2.13.3.1 Setting a trace start event and a trace end event . . . 144
2.13.3.2 Combining multiple events [E1] [E20] [EZ Emulator] . . . 147
2.13.3.3 Executing the program . . . 147
2.13.3.4 Editing trace start and trace end events . . . 147
2.13.3.5 Deleting a trace start or trace end event . . . 148
2.13.4 Collecting an execution history only when conditions are met . . . 148
2.13.5 Stopping/Restarting Collection of Execution History [E20] [Simulator] . . . 149
2.13.5.1 To temporarily stop collection of execution history . . . 149
2.13.5.2 To restart collection of execution history . . . 151
2.13.6 Displaying an execution history . . . 151
2.13.6.1 Changing the display mode . . . 151
2.13.6.2 Changing the form in which values are displayed . . . 152
2.13.6.3 Getting linked to other panels . . . 152
2.13.7 Clearing the trace memory . . . 152
2.13.8 Searching for trace data . . . 153
2.13.8.1 Searching at the instruction level . . . 153
2.13.8.2 Searching at the source level . . . 155
2.13.9 Saving the displayed content of an execution history . . . 157
2.14 Measuring the Execution Time . . . 159
2.14.1 Setting the timer measurement operation [E1] [E20] [EZ Emulator] . . . 159
2.14.2 Measuring execution time from start to stop . . . 159
2.14.2.1 Checking the status bar for confirmation . . . 160
2.14.2.2 Checking the Events panel for confirmation . . . 160
2.14.3 Measuring execution time in a section . . . 161

2.14.3.1 Setting the timer start event and timer end event . . . 161
2.14.3.2 Execute the program . . . 165
2.14.3.3 Editing a timer measurement event [E1] [E20] [EZ Emulator] . . . 167
2.14.3.4 Editing timer start and timer end events . . . 168
2.14.3.5 Deleting a timer start event or timer end event . . . 168
2.14.4 Range of measurable time . . . 168
2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]170
2.15.1 Configure the coverage measurement . . . 170
2.15.2 Display coverage measurement results . . . 171
2.16 Set an Action into Programs . . . 174
2.16.1 Insert printf . . . 174
2.16.1.1 Set a Printf event . . . 174
2.16.1.2 Execute the program . . . 175
2.16.1.3 Check the output result . . . 175
2.16.1.4 Edit Printf event . . . 175
2.16.2 Insert an interrupt event [Simulator] . . . 176
2.16.2.1 Set an interrupt event . . . 176
2.16.2.2 Execute the program . . . 177
2.16.2.3 Edit interrupt event . . . 177
2.17 Event Management . . . 178
2.17.1 Changing states of setting (Enabled/Disabled) . . . 180
2.17.2 Displaying only a specific type of event . . . 180
2.17.3 Jump to an event address . . . 180
2.17.4 Editing detailed settings of events . . . 181
2.17.4.1 Editing execution-related events . . . 181
2.17.4.2 Editing access-related events . . . 182
2.17.4.3 Editing combination conditions of events [E1] [E20] [EZ Emulator] . . . 184
2.17.5 Deleting events . . . 187
2.17.6 Entering comments for events . . . 187
2.17.7 Points to note regarding event setting . . . 187
2.17.7.1 Allowable number of valid events . . . 188
2.17.7.2 Types of events that can be set or removed during execution . . . 189
2.17.7.3 Other precautions . . . 190
2.18 Setting Up the Hook Process . . . 191
2.19 Measuring Current Consumption [E2] . . . 194
2.20 Using the Debug Console . . . 200
2.21 About Input Value . . . 204
2.21.1 Input rule . . . 204
2.21.2 Symbol name completion function . . . 208
2.21.3 Icons for invalid input . . . 209
2.22 Differences between the Microcontroller and the Emulator . . . 210
2.23 Other Notes on Usage [E1] [E20] [EZ Emulator] . . . 211

A. WINDOW REFERENCE 212
A.1 Description 212
B. I/O FUNCTIONS 407
B.1 Standard I/O and File I/O 407
Revision Record . . . . . C - 1

1. GENERAL

CS+ is an integrated developing environment platform for RH850 family, RX family, V850 family, RL78 family, 78K0R microcontroller and 78K0 microcontroller.

CS+ 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.

Caution When the E2 emulator Lite (abbreviated name: E2 Lite) is used, please read references to "E1" in this manual as also meaning "E2 Lite".
When the E2 emulator (abbreviated name: E2) is used, please read references to "E1" in this manual as also meaning "E2".

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 CS+.

1.2 Features

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

- Connectable to various debugging tools
  Usable in combination with the on-chip debugging emulator (E1/E20/EZ Emulator) 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.

2. FUNCTIONS

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

2.1 Overview

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

(1) Start CS+
Launch CS+ from the [Start] menu of Windows®.

Remark For details on "Start CS+", see "CS+ Integrated Development Environment User's Manual: Project Operation".

(2) Set a project
Create a new project, or load the existing one.

Remark For details on "Set a project", see "CS+ Integrated Development Environment User's Manual: Project Operation".

(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 "CS+ Integrated Development Environment User's Manual: CC-RX Build Tool Operation".

(4) Confirm the connection to a host machine
Connect the debug tool (E1, E20, EZ Emulator 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]
- [EZ Emulator]
- [Simulator]

(7) Connect the debug tool to CS+
Connect the debug tool to CS+ 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] [EZ Emulator]").

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] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]

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.

Remark 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

Remark 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 CS+
Disconnect the debug tool from CS+ 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 "CS+ Integrated Development Environment User's Manual: Project Operation".

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

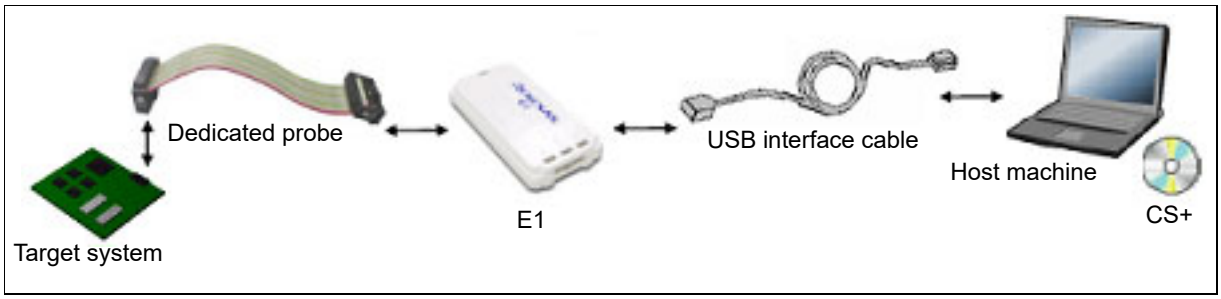
Microcontroller	Debug Tool					
	E1(Serial)	E1(JTAG)	E20(Serial)	E20(JTAG)	EZ Emulator	Simulator
RX610, RX621, RX62N, RX62T, and RX62G Groups	-	✓	-	✓	-	✓
RX630, RX631, RX63N, and RX634 Groups	✓	✓	✓	✓	-	✓
RX63T Group	✓	✓	✓	✓	✓	✓
RX110, RX111, RX113, RX210, and RX220 Groups	✓	-	✓	-	✓	✓
RX21A, RX211, RX231, RX23W, RX23T, RX24T, RX24U, RX130, RX140, RX13T and RX23E-A Groups	✓	-	✓	-	-	✓
RX71M, RX66x, RX64M, RX65N, RX651, RX65N-2M, RX651-2M, RX671, RX72T, RX72M and RX72N Groups	✓	✓	✓	✓	-	✓

2.2.1.1 [E1]
2.2.1.2 [E20]
2.2.1.3 [EZ Emulator]
2.2.1.4 [Simulator]

2.2.1.1 [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]



2.2.1.2 [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)]

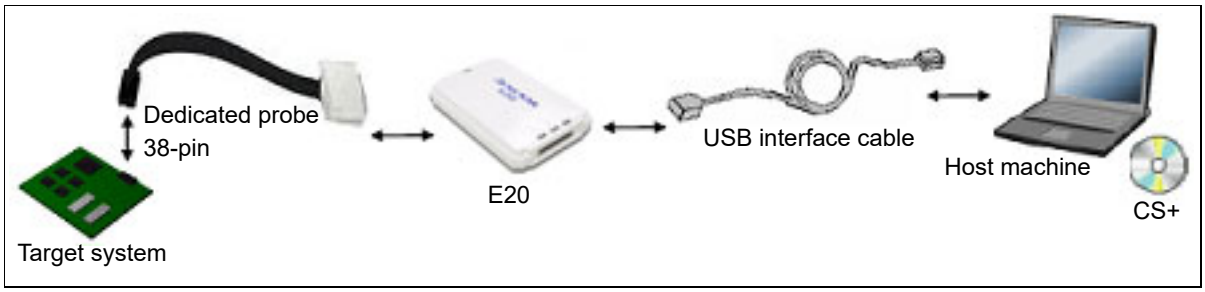
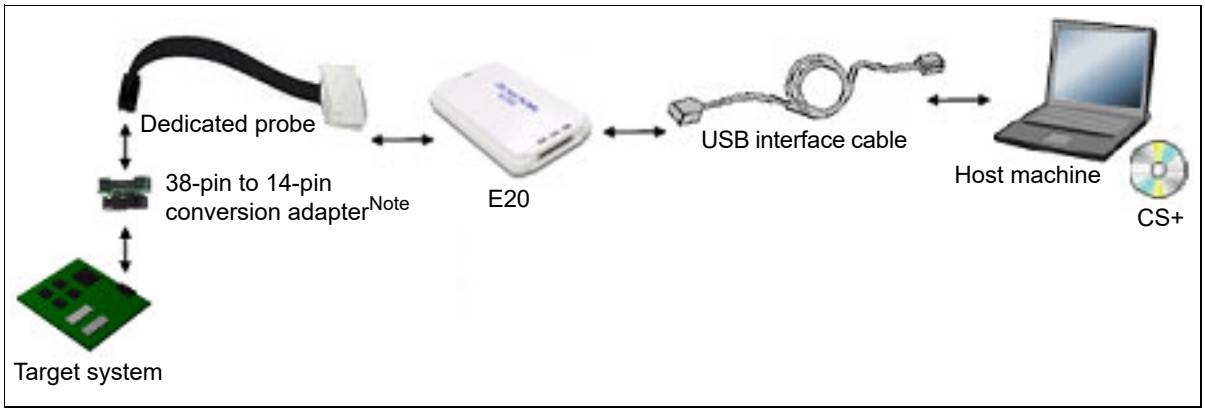


Figure 2.3 Connection Example [E20(Serial)]

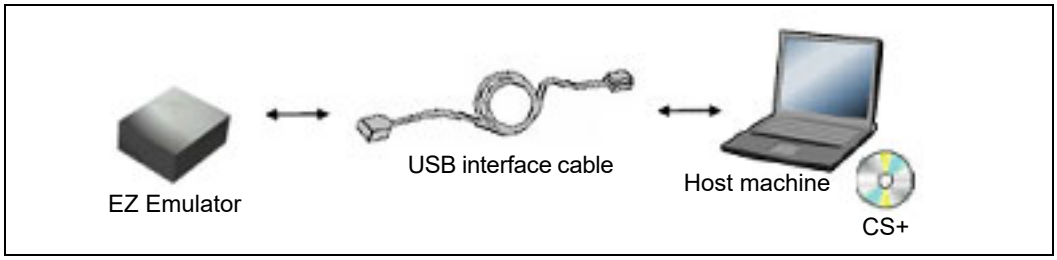


Note Use a 38-pin to 14-pin conversion adapter attached to the E20 emulator if you have selected FINE [E20 (Serial)] as the method of communication with the target system.
For more details on the connection using a 38-pin to 14-pin conversion adapter, see E20 User's Manual

2.2.1.3 [EZ Emulator]

Connect a host machine and EZ Emulator. If required, connect a target board, too.
See EZ Emulator User's Manual for details on the connection method.

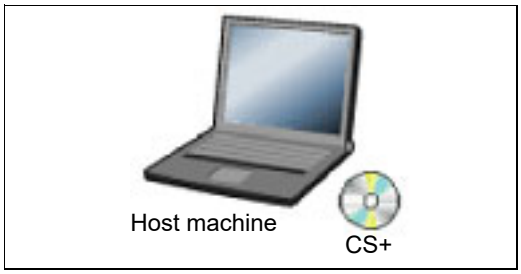
Figure 2.4 Connection Example [EZ Emulator]



2.2.1.4 [Simulator]

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

Figure 2.5 Connection Example [Simulator]



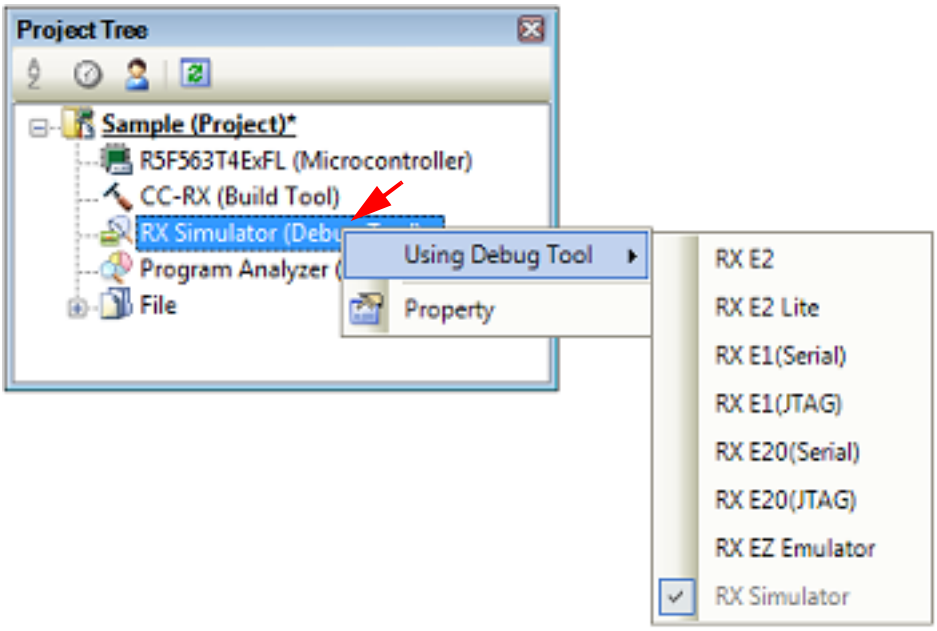
2.3 Configuration of Operating Environment of the Debug Tool

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 2.6 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 1. **[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.

Remark 2. **[EZ Emulator]**
Only FINE is supported for communication between the EZ Emulator 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 2.7 Property Panel [E1(Serial)]

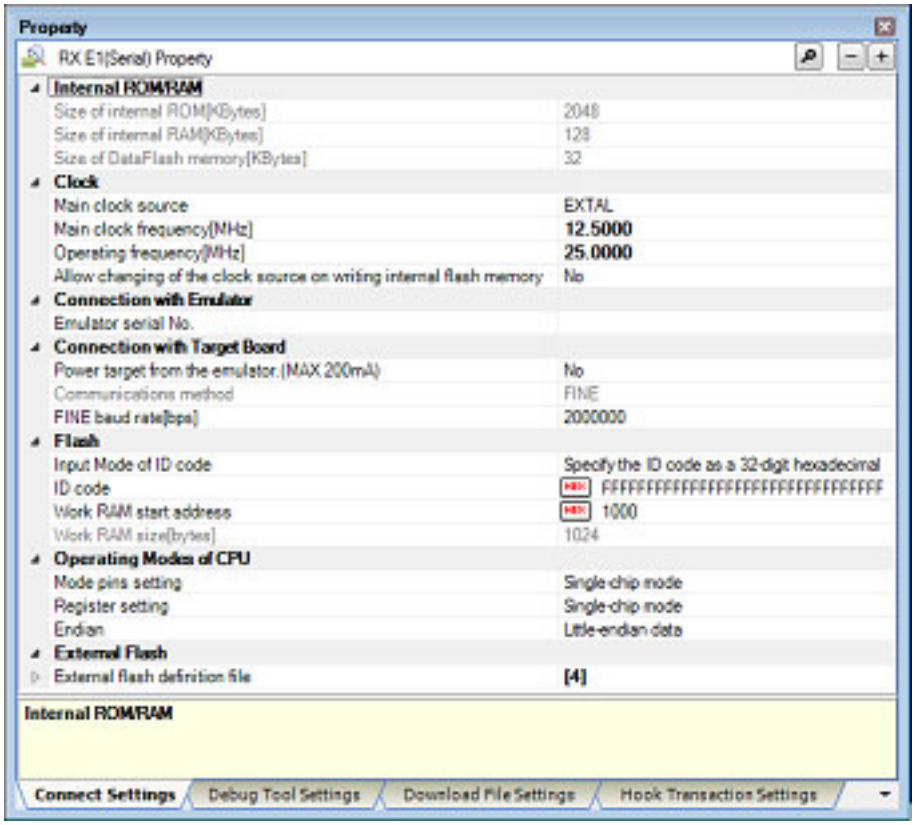
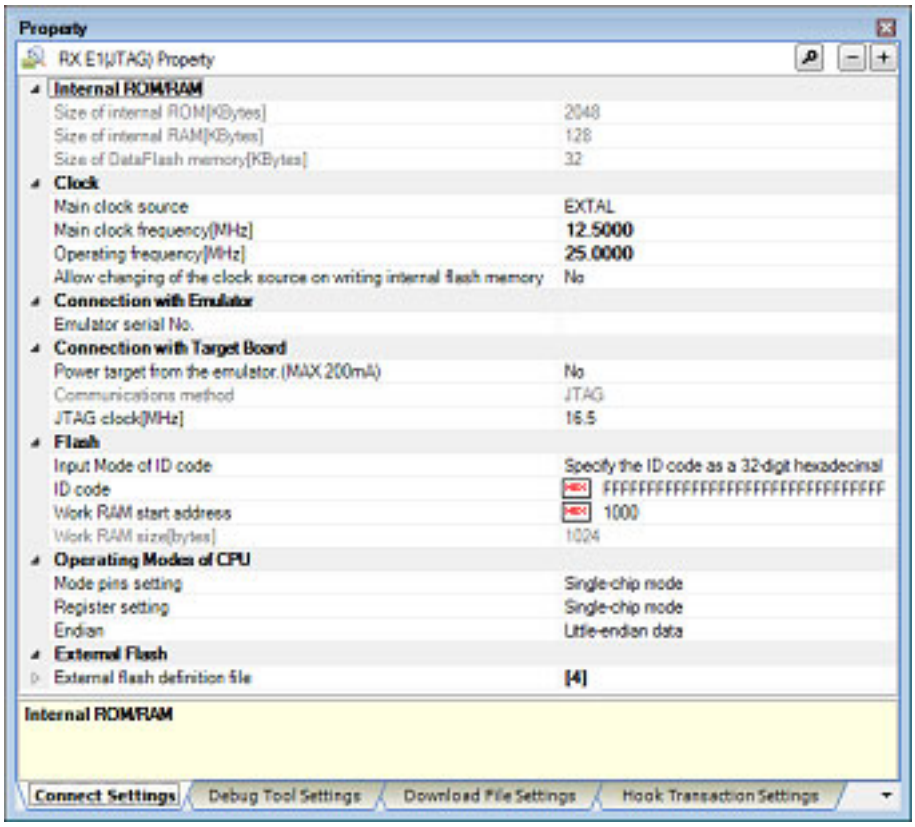


Figure 2.8 Property Panel [E1(JTAG)]



Follow the steps below by selecting the corresponding tab on the Property panel.

2.3.2.1 [Connect Settings] tab
2.3.2.2 [Debug Tool Settings] tab
2.3.2.3 [Download File Settings] tab
2.3.2.4 [Hook Transaction Settings] tab

2.3.2.1 [Connect Settings] tab

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

(1) [Internal ROM/RAM]
(2) [Clock]
(3) [Connection with Emulator]
(4) [Connection with Target Board]
(5) [E2 Expansion Interface] [E2]
(6) [Flash]
(7) [Operating Modes of CPU]
(8) [External Flash]

(1) [Internal ROM/RAM]
The configuration of internal ROM/RAM is displayed in this category.

Figure 2.9 [Internal ROM/RAM] Category

(a) [Size of internal ROM[KBytes]
The internal ROM size to emulate is displayed (unit: Kbytes).

You cannot change the value of this property.

(b) [Size of internal RAM[Bytes]]
The internal RAM size to emulate is displayed (unit: bytes).
You cannot change the value of this property.

(c) [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.

(2) [Clock]
You can configure the clock in this category.

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

Figure 2.10 [Clock] Category [HOCO]

Clock	
Main clock source	HOCO
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

Figure 2.11 [Clock] Category [EXTAL]

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

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

(b) [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 4 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.

(c) [Operating frequency [MHz]]
Specify the Operating frequency (ICLK) by directly entering a number between 0.0001 and 999.999 (MHz).
If the value is out of the specifiable range, it will be rounded to 0.0001 (if 0 or negative) or to 999.999 (if 1000 or greater).

(d) [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.

Caution [E1]
When [Yes] is selected, if internal flash ROM is rewritten by the debugger while the FlashIF clock (FCLK) of the microcontroller is outside of the guaranteed operating range (that is, while operating with LOCO or subclock), the E1 will switch the clock source. After rewriting to the internal flash ROM is completed, the clock will be restored to the previous clock source.
Note that the operating frequency of the peripheral clock will change during internal flash memory rewriting because the clock source is switched.
The clock manipulation enabling setting takes effect when the internal flash ROM is rewritten after program execution or step execution. Note that the clock source is forcibly switched regardless of the clock manipulation enabling setting if FCLK is outside of the guaranteed operating range immediately after the debug tool is activated or when the [CPU Reset] button is clicked.

(3) [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 2.12 [Connection with Emulator] Category

Connection with Emulator	
Emulator serial No.	E1: XXXXXXXXX

(a) [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.

(4) [Connection with Target Board]
You can configure the connection between E1 and the target board in this category.

Figure 2.13 [Connection with Target Board] Category [E1(Serial)]

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

Figure 2.14 [Connection with Target Board] Category [E1(JTAG)]

Connection with Target Board	
Power target from the emulator.(MAX 200mA)	Yes
Supply voltage[V]	3.3V
Communications method	JTAG
JTAG clock[MHz]	16.5

(a) [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.

(b) [Interface for supplying the power] [E2]
This property appears only when the [Power target from the emulator. (MAX 200mA)] property is set to [Yes].
Select the interface for supplying the power to the target board from the emulator (default: [USER I/F]).

(c) [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 [E1]
This property cannot be changed while connected to E1.
[E2 Lite]
This property cannot be changed.

(d) [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 when specifying E1. When specifying [RX E2 Lite] or [RX E2], whether this property can be changed differs with the type of the selected microcontroller and the communication method. See "2.2.1 Confirm the connection to a host machine" for the relationship between types of the microcontroller and the communication method.
For the details of debug tool selection, see "2.3.1 Select the debug tool to use".

(e) [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 [E1]
- 6.00 (default), 3.00, 1.50 [E2 Lite]
- 16.5 (default), 12.5, 6.25, 3.125, 1.562 [E2]

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

Caution 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.

(f) [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 [E1][E2]
- 1500000 (default), 750000, 500000, 250000 [E2 Lite]

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

Caution 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 achieve successful communication.

(5) [E2 Expansion Interface] [E2]
You can configure the E2 expansion interface.

Figure 2.15 [E2 Expansion Interface] Category [E2]

E2 Expansion Interface	
Use the E2 expansion interface	No use

(a) [Interface for supplying the power]
Select whether to use the E2 expansion interface.
If you want to use the E2 expansion interface, select the power used by the E2 expansion interface.

(6) [Flash]
You can configure the flash memory rewriting in this category.

Caution This property cannot be changed while connected to E1.

Figure 2.16 [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

(a) [Input Mode of ID code]
Specify a mode for the input of ID code.

Caution This property cannot be changed while connected to E1.

(b) [ID code]
Enter the ID code to release the flash memory from the protected state.
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.

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

Caution 2. To enter the ID code as a 32-digit hexadecimal value, arrange it as a sequence of 32-bit units of data.
Byte order of the ID code of the flash programmer can be different from that of CS+. Please confirm the specification about ID code of the flash programmer you're using.

Caution 3. If the ID code entered in ASCII characters is shorter than 16 characters, the unused space will be padded with 0.

Caution 4. Even if you have downloaded a program that contains an ID code, that ID code is replaced with FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF.

(c) [Work RAM start address]
Specify the location address of the work RAM to be used by the debugger. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected. The amount of memory indicated by the [Work RAM size[bytes] property beginning with this address is to be used by the debugger firmware.

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

Caution 2. The work RAM area can also be used by the user program because the emulator saves and restores data in this area. Note, however, that the work RAM area is not specifiable as:
the destination or origin of a DMA or DTC transfer, an address where a DTC vector table or transfer information is to be allocated, or the interrupt vector for a DMAC or DTC activation source.

(d) [Work RAM size[bytes]
Displays the size of the work RAM to be used by the debugger.

(7) [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 2.17 [Operating Modes of CPU] Category

(a) [Mode pins setting]
Specify the operating mode set by the mode pin of the microcontroller.

(b) [Allow erasing the USB boot program]
Select whether to erase the USB boot program in the user boot area when you are starting up the emulator in the user boot mode by using a microcontroller in which the USB boot program is stored.
This property is displayed only when [User boot mode] is selected for the [Mode pins setting] property.
Note that this is not displayed for an microcontroller in which the USB boot program in the user boot area does not need to be erased when you are starting up the emulator.

(c) [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.

(d) [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.

(e) [Change the start bank][RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups]
Specify whether to change the start bank before connecting to the debug tool.
Select [Yes] to change the start bank before connecting to the debug tool ([No] is selected by default).

(f) [Start bank][RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups]
Specify the start bank to be changed before connecting to the debug tool.
This property is displayed only when [Yes] is selected for the [Change the start bank][RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups] property.

(8) [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 "2.5.2.5 Downloading files to external flash memory [E1] [E20]".

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

2.3.2.2 [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.

(1) [Memory]
(2) [Access Memory While Running]
(3) [Register]
(4) [E2 Expansion Interface] [E2]
(5) [Break]
(6) [System]
(7) [Trace]
(8) [Timer] [RX600, RX700 series]

(1) [Memory]
You can configure the memory in this category.

Figure 2.18 [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

(a) [Memory mappings]
Current memory mapping status is displayed for each type of memory area.
It is not possible to change mapping on this panel.
To add or delete an I/O protection area, select the [Memory mappings] property and click on the [...] button that appears on the right to open the Memory Mapping dialog box. For details on how to change settings, refer to the section of the Memory Mapping dialog box.
This property displays only the number of memory areas.
Expanding the [Memory mappings] 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[Note 1] and data flash[Note 2]
On-chip RAM area	On-chip RAM[Note 4]
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[Note 3], FCU firmware[Note 3], user boot
Reserved area	Areas other than those listed above
I/O protection area	Address range in an external area that is not read by the debugger. Register this area in the Memory Mapping dialog box.

Note 1. If data in the on-chip flash ROM area is changed by means other than downloading (e.g. by manipulating it via the Memory panel or line assembling), the flash ROM reflects this change next time the user program is run.

Note 2. If an attempt is made to reference data in erased data flash ROM, only undefined values are displayed due to the specifications of the microcontroller. If the debugger is used to write to the data flash ROM, on the other hand, data is written in 256-byte units. Written areas do not hold undefined values.

Note 3. Do not use the debugger to write to FCU-RAM. The FCU firmware area also cannot be written by the debugger.

Note 4. The area including the option setting memory areas listed below is displayed as an on-chip RAM area. Note that values cannot be specified in this area through the memory panel.

[RX71M, RX66x, RX65x, RX64M, and RX72T Groups]

- SPCC Serial Programmer Command Control Reg.
- OSIS OCD/Serial Programmer ID Setting Reg.
- Endian select registers (MDE)
- Option function select register 0 (OFS0)
- Option function select register 1 (OFS1)

[RX65N-2M, RX651-2M, RX671, RX72M, RX66N, and RX72N]

- SPCC Serial Programmer Command Control Reg.
- OSIS OCD/Serial Programmer ID Setting Reg.
- Endian select registers (MDE)
- Option function select register 0 (OFS0)
- Option function select register 1 (OFS1)
- Bank Select Register (BANKSEL)
- Flash Access Window Setting Register (FAW)

- [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 dis-

connected.

- [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 CS+") will display details for each memory type.

(b) [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.

(2) [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 "2.11.1.4 Displaying and changing memory contents during program execution" for details on the real-time display update function.

Figure 2.19 [Access Memory While Running] Category

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

(a) [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).

(b) [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).

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

(c) [Display update interval[ms]]
This property is displayed 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).
Note that if you've changed the specified value of the [Update the display during execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(3) [Register]
In this category, make settings related to PC display in the Status bar during program execution.

Figure 2.20 [Register] Category

Register	
PC display during the execution	Yes
Display update interval for PC[ms]	500

(a) [PC display during the execution]
This property specifies whether the PC value is displayed in the Status bar during program execution.
When you select [No], the Status bar under execution will show "Running."

Caution 1. You cannot change this property while the program is in execution.

Caution 2. [RX100 Series]
This property is hidden because these microcontrollers do not support display of the PC value in the status bar during program execution.

Caution 3. While current consumption is being measured, this property is fixed to [No].

(b) [Display update interval for PC[ms]]
This property is displayed only when you've selected [Yes] in the [PC display during the execution] property.

During program execution, specify a PC display updating interval in the Status bar in 100 ms units.
Enter an integer directly in the range 100 to 65500 (with fractions below 100 ms rounded up). (By default, [500] is specified.)
Note that if you've changed the specified value of the [PC display during the execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(4) [E2 Expansion Interface] [E2]
You can configure the E2 Expansion Interface.

Figure 2.21 [E2 Expansion Interface] Category

E2 Expansion Interface	
External trigger input	**[2]**
[0]	Ch0 - Rising Edge - Stop program
Channel number	0
Use	**Yes**
Input signal	Rising Edge
Action when inputting the external trigger	Stop program
[1]	Ch1 - Rising Edge - Stop program
Channel number	1
Use	**Yes**
Input signal	Rising Edge
Action when inputting the external trigger	Stop program
External trigger output	**[2]**
[0]	Ch0 - Stop program - High Pulse - 1
Channel number	0
Use	**Yes**
Output timing	Stop program
Output signal	High Pulse
Pulse width [us]	1
[1]	Ch1 - Stop program - High Pulse - 1
Channel number	1
Use	**Yes**
Output timing	Stop program
Output signal	High Pulse
Pulse width [us]	1

(a) [External trigger input]
Set the settings related to the external trigger input.
You can select different actions for each channel.

<1> Channel number
The channel number is displayed.

<2> Use
Specify whether to use the external trigger input for this channel number.

<3> Input signal
Specify the input signal.

<4> Action when inputting the external trigger
The action when inputting the external trigger is displayed.

(b) [External trigger output]
Set the settings related to the external trigger output.
You can select different actions for each channel.

<1> Channel number
The channel number is displayed.

<2> Use
Specify whether to use the external trigger output for this channel number.

<3> Output timing
The output timing is displayed.

<4> Output signal

The output signal displayed.

<5> Pulse width [us]
Specify the pulse width.

(5) [Break]
You can configure the break function in this category.

Figure 2.22 [Break] Category

Break	
Type of breakpoints to be preferentially used	Hardware break

(a) [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).

(6) [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] [EZ Emulator]".

Figure 2.23 [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	
Work RAM start address for executing a specified routine	HEX **1DD0**
Work RAM size [bytes] for executing a specified routine	560

(a) [Debug the program re-writing the on-chip PROGRAM ROM]
Specify whether to debug programs that rewrite on-chip program ROM area, such as those that use ROM P/E mode.

Caution You cannot change this property while connected to E1.

(b) [Debug the program re-writing the on-chip DATA FLASH]
Specify whether to debug programs that rewrite on-chip data flash area, such as those that use data flash P/E mode.

Caution You cannot change this property while connected to E1.

(c) [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.

(d) [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.

(e) [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.

(f) [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.

(g) [Work RAM start address for executing a specified routine]
Specify the address where the work RAM for use in execution of the specified routine starts. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected. The amount of memory indicated by the [Work RAM size [bytes] for executing a specified routine] property beginning with this address is to be used by the debugger firmware.
This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.

Caution 1. You cannot change this property while the program is in execution.

Caution 2. Specify the range of memory that is not used by the user program.

(h) [Work RAM size [bytes] for executing a specified routine]
Indicates the size of the work RAM for use in execution of the specified routine.
This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.

(7) [Trace]
You can configure the trace function in this category.

Figure 2.24 [Trace] Category

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	**Branch + Data access**
Bus master of data access	CPU
Output timestamp	**Yes**
Trace clock count source[MHz]	
Division ratio of trace clock count source	1/1

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

(b) [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.

(c) [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, RX700 Series]
  Branch, Branch + Data access, Data access
- [RX100, RX200 Series]
  Branch, Data access

(d) [Bus master of data access][RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups]
This property is displayed only when [Branch + Data access] or [Data access] is specified in the [Trace data type] property.
Select the Bus master of data access from the drop-down list.
The following bus masters are displayed in the drop-down list.

- CPU (default), DMAC/DTC

For data access tracing, only the trace results of data access from the specified bus master are displayed on the trace panel.

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for selecting the Bus master of data access, the [Bus master of data access][RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups] property is not displayed. In this case, the bus master is fixed to [CPU].

(e) [Output timestamp]
Specify whether timestamp information is added to the trace data to be collected.
The change options in this property vary depending on the microcontroller series and the specified value of [Trace data type] property, as follows:

Microcontroller	Data type	Change options
RX600, RX700 Series	Branch, Branch + Data access, Data access	Changeable only when program is halted.
RX200 Series	Branch	Not changeable. [No] is always displayed.
	Data access	Changeable only when program is halted.
RX100 Series	Branch, Data access	Not changeable. [No] is always displayed.

(f) [Trace clock count source[MHz]]
This property is displayed only when you've specified [Yes] in the [Output timestamp] property.
Enter a count source with which a timestamp value is calculated from a count value.
To specify this, enter a value directly in the range 0.0001 to 999.999.
Note that if this property is blank, the set value of the [Operating frequency [MHz]] property in the [Clock] category of a [Connect Settings] tab is used in place of the count source.

Caution 1. You cannot change this property while the program is in execution.

Caution 2. The property's set value is not reflected in the trace data that has already been collected. The property you've set is reflected beginning with the trace data that is collected after it is set.

(g) [Division ratio of trace clock count source][RX71M, RX66x, RX65x, and RX64M Groups]
This property is displayed only when [Yes] is selected in the [Output timestamp] property.
Select the frequency division ratio for the timestamp count source from the drop-down list.
The following frequency division ratios are displayed in the drop-down list.

- 1/1 (default), 1/16, 1/256, 1/4096

The frequency specified in the [Trace clock count source[MHz]] property is divided by the specified value (the frequency is multiplied by 1/n) and one cycle of the obtained frequency is used as the unit for timestamp count (the frequency for count value 1).

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for dividing the timestamp frequency for tracing, the [Division ratio of trace clock count source][RX71M, RX66x, RX65x, and RX64M Groups] property is not displayed. In this case, the division ratio is fixed to [1/1].

(8) [Timer] [RX600, RX700 series]
You can configure the timer function in this category.

Figure 2.25 [Timer] Category

(a) [Use 64bit counter]
Specify whether to use two 32-bit counters or one 64-bit counter.

2.3.2.3 [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".

2.3.2.4 [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 2.26 Property Panel [E20(Serial)]

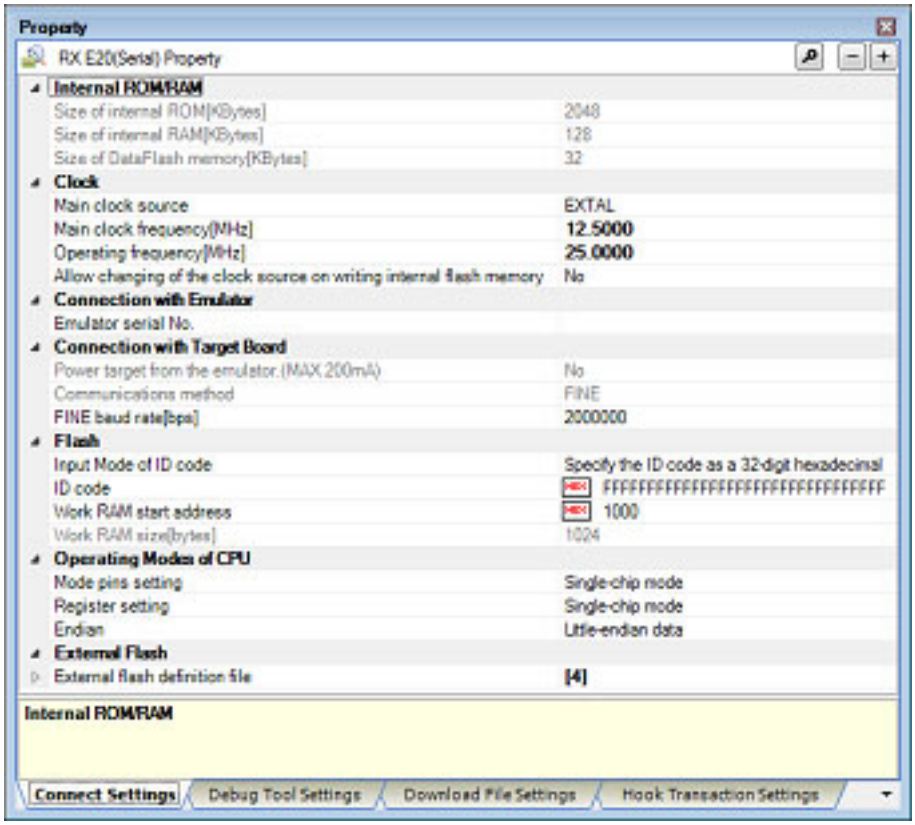
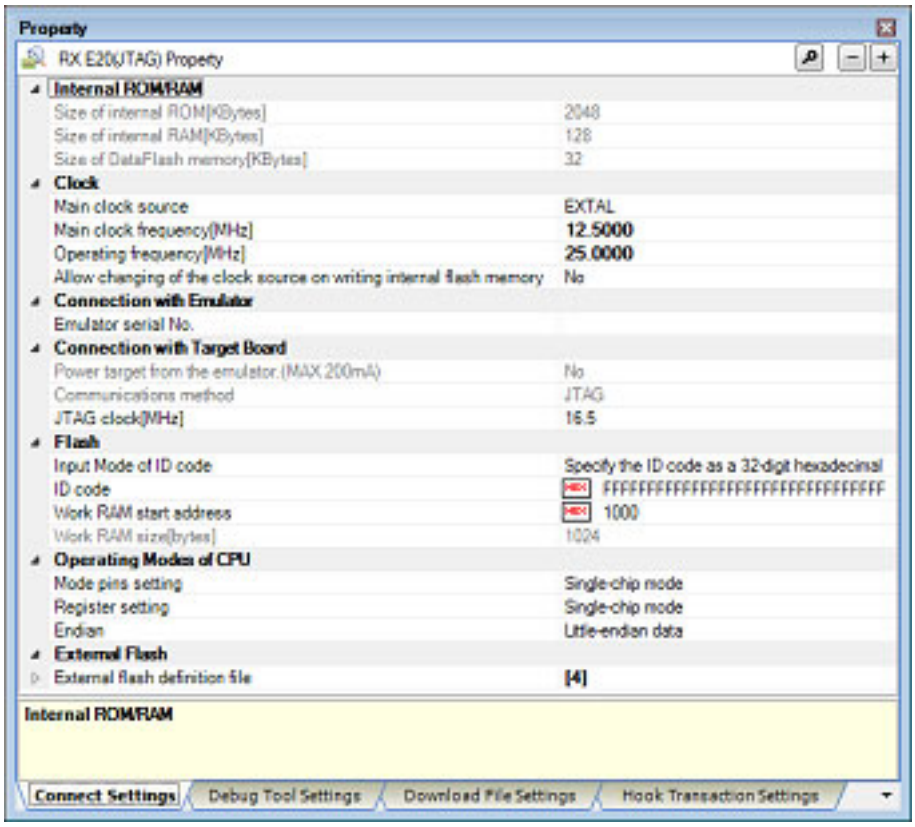


Figure 2.27 Property Panel [E20(JTAG)]



Follow the steps below by selecting the corresponding tab on the Property panel.

2.3.3.1 [Connect Settings] tab
2.3.3.2 [Debug Tool Settings] tab
2.3.3.3 [Download File Settings] tab
2.3.3.4 [Hook Transaction Settings] tab

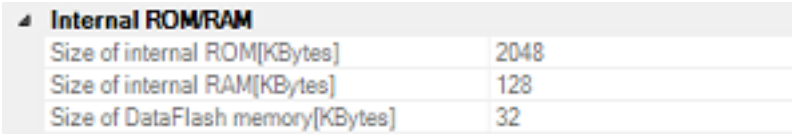
2.3.3.1 [Connect Settings] tab

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

(1) [Internal ROM/RAM]
(2) [Clock]
(3) [Connection with Emulator]
(4) [Connection with Target Board]
(5) [Flash]
(6) [Operating Modes of CPU]
(7) [External Flash]

(1) [Internal ROM/RAM]
The configuration of internal ROM/RAM is displayed in this category.

Figure 2.28 [Internal ROM/RAM] Category



(a) [Size of internal ROM[KBytes]
The internal ROM size to emulate is displayed (unit: Kbytes).
You cannot change the value of this property.

(b) [Size of internal RAM[Bytes]]
The internal RAM size to emulate is displayed (unit: bytes).
You cannot change the value of this property.

(c) [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.

(2) [Clock]
You can configure the clock in this category.

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

Figure 2.29 [Clock] Category [HOCO]

Clock	
Main clock source	HOCO
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

Figure 2.30 [Clock] Category [EXTAL]

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

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

(b) [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 4 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.

(c) [Operating frequency [MHz]]
Specify the Operating frequency (ICLK) by directly entering a number between 0.0001 and 999.999 (MHz).
If the value is out of the specifiable range, it will be rounded to 0.0001 (if 0 or negative) or to 999.999 (if 1000 or greater).

(d) [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.

Caution [E20]
When [Yes] is selected, if internal flash ROM is rewritten by the debugger while the FlashIF clock (FCLK) of the microcontroller is outside of the guaranteed operating range (that is, while operating with LOCO or subclock), the E20 will switch the clock source. After rewriting to the internal flash ROM is completed, the clock will be restored to the previous clock source.
Note that the operating frequency of the peripheral clock will change during internal flash memory rewriting because the clock source is switched.
The clock manipulation enabling setting takes effect when the internal flash ROM is rewritten after program execution or step execution. Note that the clock source is forcibly switched regardless of the clock manipulation enabling setting if FCLK is outside of the guaranteed operating range immediately after the debug tool is activated or when the [CPU Reset] button is clicked.

(3) [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 2.31 [Connection with Emulator] Category

Connection with Emulator	
Emulator serial No.	E20:XXXXXXXXX

(a) [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.

(4) [Connection with Target Board]
You can configure the connection between E20 and the target board in this category.

Figure 2.32 [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 2.33 [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

(a) [Power target from the emulator. (MAX 200mA)]
[No] is displayed as the property value.
E20 does not support power supply function.

(b) [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".

(c) [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

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

Caution 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.

(d) [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

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

Caution 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.

(5) [Flash]
You can configure the flash memory rewriting in this category.

Figure 2.34 [Flash] Category

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

(a) [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.

(b) [ID code]
Enter the ID code to release the flash memory from the protected state.
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.

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

Caution 2. To enter the ID code as a 32-digit hexadecimal value, arrange it as a sequence of 32-bit units of data.
Byte order of the ID code of the flash programmer can be different from that of CS+. Please confirm the specification about ID code of the flash programmer you're using.

Caution 3. If the ID code entered in ASCII characters is shorter than 16 characters, the unused space will be padded with 0.

Caution 4. Even if you have downloaded a program that contains an ID code, that ID code is replaced with FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF.

(c) [Work RAM start address]
Specify the location address of the work RAM to be used by the debugger. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected.
The amount of memory indicated by the [Work RAM size[bytes]] property beginning with this address is to be used by the debugger firmware.

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

Caution 2. The work RAM area can also be used by the user program because the emulator saves and restores data in this area. Note, however, that the work RAM area is not specifiable as:
the destination or origin of a DMA or DTC transfer, an address where a DTC vector table or transfer information is to be allocated, or the interrupt vector for a DMAC or DTC activation source.

(d) [Work RAM size[bytes]]
Displays the size of work RAM to be used by the debugger.

(6) [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 2.35 [Operating Modes of CPU] Category

Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data
Change the start bank	Yes
Start bank	Bank 0

(a) [Mode pins setting]
Specify the operating mode set by the mode pin of the microcontroller.

(b) [Allow erasing the USB boot program]
Select whether to erase the USB boot program in the user boot area when you are starting up the emulator in the user boot mode by using a microcontroller in which the USB boot program is stored.
This property is displayed only when [User boot mode] is selected for the [Mode pins setting] property. Note that this is not displayed for an microcontroller in which the USB boot program in the user boot area does not need to be erased when you are starting up the emulator.

(c) [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.

(d) [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.

(e) [Change the start bank][RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups]
Specify whether to change the start bank before connecting to the debug tool.
Select [Yes] to change the start bank before connecting to the debug tool ([No] is selected by default).

(f) [Start bank][RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups]
Specify the start bank to be changed before connecting to the debug tool.
This property is displayed only when [Yes] is selected for the [Change the start bank][RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups] property.

(7) [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 "2.5.2.5 Downloading files to external flash memory [E1] [E20]".

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

2.3.3.2 [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.

(1) [Memory]
(2) [Access Memory While Running]
(3) [Register]
(4) [Break]
(5) [System]
(6) [Trace]
(7) [Timer] [RX600, RX700 series]
(8) [Coverage] [RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups]

(1) [Memory]
You can configure the memory in this category.

Figure 2.36 [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

(a) [Memory mappings]
Current memory mapping status is displayed in detail for each type of memory area.
It is not possible to change mapping on this panel.
To add or delete an I/O protection area, select the [Memory mappings] property and click on the [...] button that appears on the right to open the Memory Mapping dialog box. For details on how to change settings, refer to the section of the Memory Mapping dialog box.
This property displays only the number of memory areas.
Expanding the [Memory mappings] 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[Note 1] and data flash[Note 2]
On-chip RAM area	On-chip RAM[Note 4]
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[Note 3], FCU firmware[Note 3], user boot
Reserved area	Areas other than those listed above

I/O protection area	Address range in an external area that is not read by the debugger. Register this area in the Memory Mapping dialog box.

Note 1. If data in the on-chip flash ROM area is changed by means other than downloading (e.g. by manipulating it via the Memory panel or line assembling), the flash ROM reflects this change next time the user program is run.

Note 2. If an attempt is made to reference data in erased data flash ROM, only undefined values are displayed due to the specifications of the microcontroller. If the debugger is used to write to the data flash ROM, on the other hand, data is written in 256-byte units. Written areas do not hold undefined values.

Note 3. Do not use the debugger to write to FCU-RAM. The FCU firmware area also cannot be written by the debugger.

Note 4. The area including the option setting memory areas listed below is displayed as an on-chip RAM area. Note that values cannot be specified in this area through the memory panel.

[RX71M, RX66x, RX65x, RX64M, and RX72T Groups]

- SPCC Serial Programmer Command Control Reg.
- OSIS OCD/Serial Programmer ID Setting Reg.
- Endian select registers (MDE)
- Option function select register 0 (OFS0)
- Option function select register 1 (OFS1)

[RX65N-2M, RX651-2M, RX671, RX72M, RX66N, and RX72N]

- SPCC Serial Programmer Command Control Reg.
- OSIS OCD/Serial Programmer ID Setting Reg.
- Endian select registers (MDE)
- Option function select register 0 (OFS0)
- Option function select register 1 (OFS1)
- Bank Select Register (BANKSEL)
- Flash Access Window Setting Register (FAW)

- [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 CS+") will display details for each memory type.

(b) [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.

(2) [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 "2.11.1.4 Displaying and changing memory contents during program execution" for details on the real-time display update function.

Figure 2.37 [Access Memory While Running] Category [E20(Serial)]

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

Figure 2.38 [Access Memory While Running] Category [E20(JTAG)[RX600, RX700 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

(a) [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).

(b) [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).

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

(c) [Update interval[ms]]
This property is displayed 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).
Note that if you've changed the specified value of the [Update the display during execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(d) [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.

(3) [Register]
In this category, make settings related to PC display in the Status bar during program execution.

Figure 2.39 [Register] Category

Register	
PC display during the execution	**Yes**
Display update interval for PC[ms]	500

(a) [PC display during the execution]
This property specifies whether the PC value is displayed in the Status bar during program execution.
When you select [No], the Status bar under execution will show "Running."

Caution 1. You cannot change this property while the program is in execution.

Caution 2. [RX100 Series]
This property is hidden because these microcontrollers do not support display of the PC value in the status bar during program execution.

(b) [Display update interval for PC[ms]]
This property is displayed only when you've selected [Yes] in the [PC display during the execution] property.
During program execution, specify a PC display updating interval in the Status bar in 100 ms units.
Enter an integer directly in the range 100 to 65500 (with fractions below 100 ms rounded up). (By default, [500] is specified.)

Note that if you've changed the specified value of the [PC display during the execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(4) [Break]
You can configure the break function in this category.

Figure 2.40 [Break] Category

Break	
Type of breakpoints to be preferentially used	Hardware break

(a) [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).

(5) [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] [EZ Emulator]".

Figure 2.41 [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	
Work RAM start address for executing a specified routine	HEX **1DD0**
Work RAM size [bytes] for executing a specified routine	560

(a) [Debug the program re-writing the on-chip PROGRAM ROM]
Specify whether to debug programs that rewrite on-chip program ROM area, such as those that use ROM P/E mode.

Caution You cannot change this property while connected to E20.

(b) [Debug the program re-writing the on-chip DATA FLASH]
Specify whether to debug programs that rewrite on-chip data flash area, such as those that use data flash P/E mode.

Caution You cannot change this property while connected to E20.

(c) [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.

(d) [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.

(e) [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.

(f) [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.

(g) [Work RAM start address for executing a specified routine]
Specify the address where the work RAM for use in execution of the specified routine starts. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected. The amount of memory indicated by the [Work RAM size [bytes] for executing a specified routine] property beginning with this address is to be used by the debugger firmware.
This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.

Caution 1. You cannot change this property while the program is in execution.

Caution 2. Specify the range of memory that is not used by the user program.

(h) [Work RAM size [bytes] for executing a specified routine]
Indicates the size of the work RAM for use in execution of the specified routine.
This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.

(6) [Trace]
You can configure the trace function in this category.

Figure 2.42 [Trace] Category

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	**Branch + Data access (without Access data)**
Start address for trace area without access data	HEX 0
End address for trace area without access data	HEX 3FFFF
Bus master of data access	CPU
External trace output	CPU execution
Trace memory size[MByte]	1
Output timestamp	**Yes**
Trace clock count source[MHz]	
Division ratio of trace clock count source	1/1

(a) [Usage of trace function]
Specify whether to use it as real-time RAM monitor function (RRM function) 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 "2.11.1.4 Displaying and changing memory contents during program execution".
This property can be changed only when the program is not running.

Caution To use the real-time RAM monitor function, the E20 and the target board must be connected via a 38-pin JTAG cable. If the E20 and the target board are connected in any other way or via the 38-pin to 14-pin conversion adapter, select [Trace].

(b) [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, RX700 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.

(c) [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.

- [RX66x, RX65N-2M, RX651-2M, RX671, RX72M, RX72N, and RX72T Groups]
  Branch, Branch + Data access, Data access, Branch + Data access (without Access data), or Data access (without Access data)
- [RX600, RX700 Series]
  Branch, Branch + Data access, Data access
- [RX100, RX200 Series]
  Branch, Data access

Caution [E20(JTAG) [RX600, RX700 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.

(d) [Start address for trace area without access data][RX65N-2M, RX651-2M, RX66T, RX671, RX72T, RX72M, RX66N, and RX72N Group]
Specify the start address for the trace area without access data.

(e) [End address for trace area without access data][RX65N-2M, RX651-2M, RX66T, RX671, RX72T, RX72M, RX66N, and RX72N Group]
Specify the end address for the trace area without access data.

(f) [Bus master of data access] [RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups]
Select the bus master which generated the data access.
This property is displayed only when [Branch+Data access] or [Data access] is specified in the [Trace data type] property.
The following bus masters are displayed in the drop-down list.

- CPU (default), DMAC/DTC

When [Trace] is selected in the [Usage of trace function] property,
for data access tracing, only the trace results of data access from the specified bus master are displayed on the trace panel.

Caution 1. This property setting cannot be changed during program execution.

Caution 2. When [Real-time RAM Monitor] is selected in the [Usage of trace function] property, The bus master is fixed to [CPU] and cannot be changed.

Caution 3. For an microcontroller that does not have the function for selecting the Bus master of data access, the [Bus master of data access] [RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups] property is not displayed. In this case, the bus master is fixed to [CPU].

(g) [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 1. 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.

Caution 2. If Step in is executed when [CPU execution] or [Trace output] is specified and trace data is being displayed on the Trace panel, correct trace data may not always be displayed.

Caution 3. [RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups]
When this property is changed from [Do not output] to [CPU execution] or [Trace output], the timer measurement results are initialized.

(h) [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.

(i) [Output timestamp]
Specify whether timestamp information is added to the trace data to be collected.
This property is selectable only when you've specified [Trace] in the [Usage of trace function] property.
The change options in this property vary depending on the microcontroller series and the specified value of [Trace data type] property, as follows:

Microcontroller	Data type	Change options
RX66x, RX65N-2M, RX651-2M, RX671, RX72M, RX72N, and RX72T Group	Branch, Branch + Data access, Data access, Branch + Data access (without Access data), Data access (without Access data)	Changeable only when program is halted.
RX600 Series other than those listed above	Branch, Branch+Data access, Data access	Changeable only when program is halted.
RX200 Series	Branch	Not changeable. [No] is always displayed.
	Data access	Changeable only when program is halted.
RX100 Series	Branch, Data access	Not changeable. [No] is always displayed.

(j) [Trace clock count source[MHz]]
This property is displayed only when you've specified [Yes] in the [Output timestamp] property.
Enter a count source with which a timestamp value is calculated from a count value.
To specify this, enter a value directly in the range 0.0001 to 999.999.
Note that if this property is blank, the set value of the [Operating frequency [MHz]] property in the [Clock] category of a [Connect Settings] tab is used in place of the count source.

Caution 1. You cannot change this property while the program is in execution.

Caution 2. The property's set value is not reflected in the trace data that has already been collected. The property you've set is reflected beginning with the trace data that is Division ratio of trace clock count source collected after it is set.

(k) [Division ratio of trace clock count source][RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups]
Specify the division ratio of trace clock count source for the timestamp.
This property is displayed only when [Yes] is selected in the [Output timestamp] property.
Select the frequency division ratio for the timestamp count source from the drop-down list.
The following frequency division ratios are displayed in the drop-down list.

- 1/1, 1/16, 1/256, 1/4096

The frequency specified in the [Trace clock count source[MHz]] property is divided by the specified value (the frequency is multiplied by 1/n) and one cycle of the obtained frequency is used as the unit for timestamp count (the frequency for count value 1).

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for dividing the timestamp frequency for tracing, the [Division ratio of trace clock count source][RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups] property is not displayed. In this case, the division ratio is fixed to [1/1]

(7) [Timer] [RX600, RX700 series]
You can configure the timer function in this category.

Figure 2.43 [Timer] Category

Timer	
Use 64bit counter	No

(a) [Use 64bit counter]
Specify whether to use two 32-bit counters or one 64-bit counter.

(8) [Coverage] [RX71M, RX66x, RX65x, RX64M, RX671, RX72M, RX72N, and RX72T Groups]
You can configure the coverage function in this category.
See "2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]" for details on the coverage function and this category configuration.

2.3.3.3 [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".

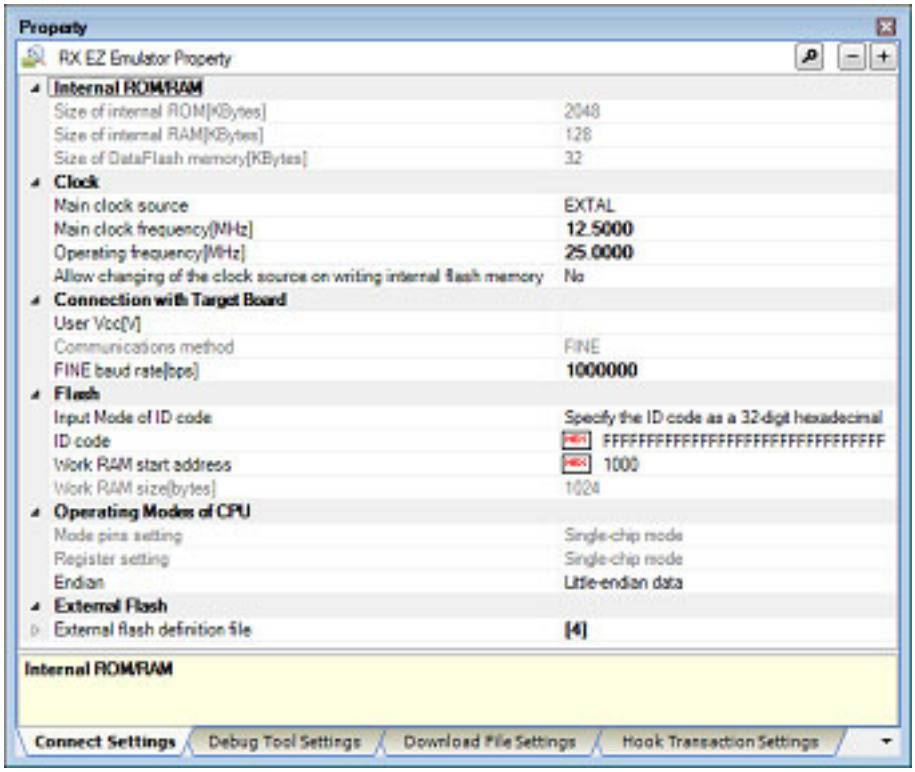
2.3.3.4 [Hook Transaction Settings] tab

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 [EZ Emulator]

Configure the operating environment on the Property panel below when using EZ Emulator.

Figure 2.44 Property Panel



Follow the steps below by selecting the corresponding tab on the Property panel.

2.3.4.1 [Connect Settings] tab
2.3.4.2 [Debug Tool Settings] tab
2.3.4.3 [Download File Settings] tab
2.3.4.4 [Hook Transaction Settings] tab

2.3.4.1 [Connect Settings] tab

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

(1) [Internal ROM/RAM]
(2) [Clock]
(3) [Connection with Target Board]
(4) [Flash]
(5) [Operating Modes of CPU]
(6) [External Flash]

(1) [Internal ROM/RAM]
The configuration of internal ROM/RAM is displayed in this category.

Figure 2.45 [Internal ROM/RAM] Category

(a) [Size of internal ROM[KBytes]
The internal ROM size to emulate is displayed (unit: Kbytes).
You cannot change the value of this property.

(b) [Size of internal RAM[Bytes]]
The internal RAM size to emulate is displayed (unit: bytes).
You cannot change the value of this property.

(c) [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.

(2) [Clock]
You can configure the clock in this category.

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

Figure 2.46 [Clock] Category [HOCO]

Clock	
Main clock source	HOCO
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

Figure 2.47 [Clock] Category [EXTAL]

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

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

(b) [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 4 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.

(c) [Operating frequency [MHz]]
Specify the Operating frequency (ICLK) by directly entering a number between 0.0001 and 999.999 (MHz).
If the value is out of the specifiable range, it will be rounded to 0.0001 (if 0 or negative) or to 999.999 (if 1000 or greater).

(d) [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.

Caution [EZ Emulator]
When [Yes] is selected, if internal flash ROM is rewritten by the debugger while the FlashIF clock (FCLK) of the microcontroller is outside of the guaranteed operating range (that is, while operating with LOCO or subclock), the EZ Emulator will switch the clock source. After rewriting to the internal flash ROM is completed, the clock will be restored to the previous clock source.
Note that the operating frequency of the peripheral clock will change during internal flash memory rewriting because the clock source is switched.
The clock manipulation enabling setting takes effect when the internal flash ROM is rewritten after program execution or step execution. Note that the clock source is forcibly switched regardless of the clock manipulation enabling setting if FCLK is outside of the guaranteed operating range immediately after the debug tool is activated or when the [CPU Reset] button is clicked.

(3) [Connection with Target Board]
You can configure the connection between EZ Emulator and the target board in this category.

Figure 2.48 [Connection with Target Board] Category

Connection with Target Board	
User Vcc[V]	
Communications method	FINE
FINE baud rate[bps]	1000000

(a) [User Vcc[V]]
Specify the value of the microcontroller's operating voltage that is actually supplied to the user board.
Directly enter a numerical value from 0.0001 to 9.9999 (unit: V).

Caution This property cannot be changed while connected to EZ Emulator.

(b) [Communications method]
Displays the method of communication used by the EZ Emulator for communicating with the microcontroller on the target system.
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".

(c) [FINE baud rate[bps]]
From the drop-down list, select the baud rate (FINE baud rate) to be used by the EZ Emulator for communicating with the microcontroller on the target system.
The following baud rate is displayed in the drop-down list.

- 1000000 (default), 500000, 250000

Caution 1. This property cannot be changed while connected to EZ Emulator.

Caution 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.

(4) [Flash]
You can configure the flash memory rewriting in this category.

Caution This property cannot be changed while connected to EZ Emulator.

Figure 2.49 [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

(a) [Input Mode of ID code]
Specify a mode for the input of ID code.

Caution This property cannot be changed while connected to EZ Emulator.

(b) [ID code]
Enter the ID code to release the flash memory from the protected state.
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.

Caution 1. This property cannot be changed while connected to EZ Emulator.

Caution 2. To enter the ID code as a 32-digit hexadecimal value, arrange it as a sequence of 32-bit units of data.
Byte order of the ID code of the flash programmer can be different from that of CS+. Please confirm the specification about ID code of the flash programmer you're using.

Caution 3. If the ID code entered in ASCII characters is shorter than 16 characters, the unused space will be padded with 0.

Caution 4. Even if you have downloaded a program that contains an ID code, that ID code is replaced with FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF.

(c) [Work RAM start address]
Specify the location address of the work RAM to be used by the debugger. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected.

The amount of memory indicated by the [Work RAM size[bytes]] property beginning with this address is to be used by the debugger firmware.

Caution 1. This property cannot be changed while connected to EZ Emulator.

Caution 2. The work RAM area can also be used by the user program because the emulator saves and restores data in this area. Note, however, that the work RAM area is not specifiable as:
the destination or origin of a DMA or DTC transfer, an address where a DTC vector table or transfer information is to be allocated, or the interrupt vector for a DMAC or DTC activation source.

(d) [Work RAM size[bytes]]
Displays the size of the work RAM to be used by the debugger.

(5) [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 EZ Emulator.

Figure 2.50 [Operating Modes of CPU] Category

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

(a) [Mode pins setting]
Specify the operating mode set by the mode pin of the microcontroller.
The setting is fixed to single-chip mode while the EZ Emulator is in use.

(b) [Register setting]
Specify the operating mode to be set by the register.
The setting is fixed to single-chip mode while the EZ Emulator is in use.

(c) [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.

(6) [External Flash]
In this category, you can configure external flash.
Although this category is controllable while the EZ Emulator is in use, any setting for this category is not reflected because the microcontroller's operating mode is fixed to single-chip mode.

2.3.4.2 [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.

(1) [Memory]
(2) [Access Memory While Running]
(3) [Register]
(4) [Break]
(5) [System]
(6) [Trace]
(7) [Timer] [RX600 series]

(1) [Memory]
You can configure the memory in this category.

Figure 2.51 [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

(a) [Memory mappings]
Current memory mapping status is displayed for each type of memory area.
It is not possible to change mapping on this panel.
To add or delete an I/O protection area, select the [Memory mappings] property and click on the [...] button that appears on the right to open the Memory Mapping dialog box. For details on how to change settings, refer to the section of the Memory Mapping dialog box.
This property displays only the number of memory areas.
Expanding the [Memory mappings] 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[Note 1] and data flash[Note 2]
On-chip RAM area	On-chip 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[Note 3], FCU firmware[Note 3], user boot
Reserved area	Areas other than those listed above

I/O protection area	Address range in an external area that is not read by the debugger. Register this area in the Memory Mapping dialog box.

Note 1. If data in the on-chip flash ROM area is changed by means other than downloading (e.g. by manipulating it via the Memory panel or line assembling), the flash ROM reflects this change next time the user program is run.

Note 2. If an attempt is made to reference data in erased data flash ROM, only undefined values are displayed due to the specifications of the microcontroller. If the debugger is used to write to the data flash ROM, on the other hand, data is written in 256-byte units. Written areas do not hold undefined values.

Note 3. Do not use the debugger to write to FCU-RAM. The FCU firmware area also cannot be written by the debugger.

- [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] Indicates the memory type of the corresponding area. Each memory type corresponds to the following areas. 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] Indicates the memory type of the corresponding area. Each memory type corresponds to the following areas. 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 CS+") will display details for each memory type.

(b) [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.

(2) [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 "2.11.1.4 Displaying and changing memory contents during program execution" for details on the real-time display update function.

Figure 2.52 [Access Memory While Running] Category

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

(a) [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).

(b) [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).

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

(c) [Update interval[ms]]
This property is displayed 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).
Note that if you've changed the specified value of the [Update the display during execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(3) [Register]
In this category, make settings related to PC display in the Status bar during program execution.

Figure 2.53 [Register] Category

Register	
PC display during the execution	Yes
Display update interval for PC[ms]	500

(a) [PC display during the execution]
This property specifies whether the PC value is displayed in the Status bar during program execution.
When you select [No], the Status bar under execution will show "Running."

Caution 1. You cannot change this property while the program is in execution.

Caution 2. [RX100 Series]
This property is hidden because these microcontrollers do not support display of the PC value in the status bar during program execution.

(b) [Display update interval for PC[ms]]
This property is displayed only when you've selected [Yes] in the [PC display during the execution] property.
During program execution, specify a PC display updating interval in the Status bar in 100 ms units.
Enter an integer directly in the range 100 to 65500 (with fractions below 100 ms rounded up). (By default, [500] is specified.)
Note that if you've changed the specified value of the [PC display during the execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(4) [Break]
You can configure the break function in this category.

Figure 2.54 [Break] Category

Break	
Type of breakpoints to be preferentially used	Hardware break

(a) [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).

(5) [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] [EZ Emulator]".

Figure 2.55 [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	
Work RAM start address for executing a specified routine	1DD0
Work RAM size [bytes] for executing a specified routine	560

(a) [Debug the program re-writing the on-chip PROGRAM ROM]
Specify whether to debug programs that rewrite on-chip program ROM area, such as those that use ROM P/E mode.

Caution You cannot change this property while connected to EZ Emulator.

(b) [Debug the program re-writing the on-chip DATA FLASH]
Specify whether to debug programs that rewrite on-chip data flash area, such as those that use data flash P/E mode.

Caution You cannot change this property while connected to EZ Emulator.

(c) [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.

(d) [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.

(e) [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.

(f) [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.

(g) [Work RAM start address for executing a specified routine]
Specify the address where the work RAM for use in execution of the specified routine starts. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected. The amount of memory indicated by the [Work RAM size [bytes] for executing a specified routine] property beginning with this address is to be used by the debugger firmware.
This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.

Caution 1. You cannot change this property while the program is in execution.

Caution 2. Specify the range of memory that is not used by the user program.

(h) [Work RAM size [bytes] for executing a specified routine]
Indicates the size of the work RAM for use in execution of the specified routine.
This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.

(6) [Trace]
You can configure the trace function in this category.

Figure 2.56 [Trace] Category

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch
Output timestamp	**Yes**
Trace clock count source[MHz]	

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

(b) [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.

(c) [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, RX700 Series]
  Branch, Branch+Data access, Data access
- [RX100, RX200 Series]
  Branch, Data access

(d) [Output timestamp]
Specify whether timestamp information is added to the trace data to be collected.
The change options in this property vary depending on the microcontroller series and the specified value of [Trace data type] property, as follows:

Microcontroller	Data type	Change options
RX600, RX700 Series	Branch, Branch+Data access, Data access	Changeable only when program is halted.
RX200 Series	Branch	Not changeable. [No] is always displayed.
	Data access	Changeable only when program is halted.
RX100 Series	Branch, Data access	Not changeable. [No] is always displayed.

(e) [Trace clock count source[MHz]]
This property is displayed only when you've specified [Yes] in the [Output timestamp] property.
Enter a count source with which a timestamp value is calculated from a count value.
To specify this, enter a value directly in the range 0.0001 to 999.999.
Note that if this property is blank, the set value of the [Operating frequency [MHz]] property in the [Clock] category of a [Connect Settings] tab is used in place of the count source.

Caution 1. You cannot change this property while the program is in execution.

Caution 2. The property's set value is not reflected in the trace data that has already been collected. The property you've set is reflected beginning with the trace data that is collected after it is set.

(7) [Timer] [RX600 series]
You can configure the timer function in this category.

Figure 2.57 [Timer] Category

Timer	
Use 64bit counter	No

(a) [Use 64bit counter]
Specify whether to use two 32-bit counters or one 64-bit counter.

2.3.4.3 [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".

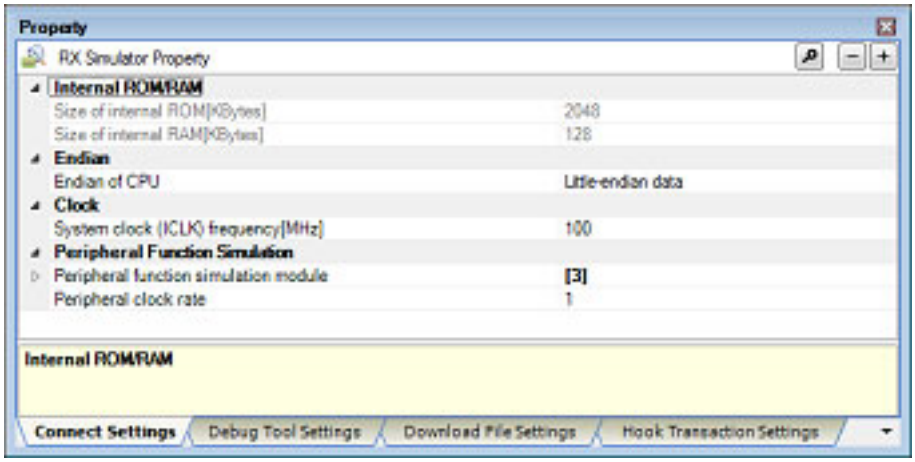
2.3.4.4 [Hook Transaction Settings] tab

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.

2.3.5 [Simulator]

Configure the operating environment on the Property panel below when using Simulator.

Figure 2.58 Operating Environment Settings [Simulator] (Property Panel)



Follow the steps below by selecting the corresponding tab on the Property panel.

2.3.5.1 [Connect Settings] tab
2.3.5.2 [Debug Tool Settings] tab
2.3.5.3 [Download File Settings] tab
2.3.5.4 [Hook Transaction Settings] tab

2.3.5.1 [Connect Settings] tab

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

(1) [Internal ROM/RAM]
(2) [Endian]
(3) [Clock]
(4) [Peripheral Function Simulation]

(1) [Internal ROM/RAM]
The size of internal ROM/RAM of the selected microcontroller is displayed in this category.

Figure 2.59 [Internal ROM/RAM] Category [Simulator]

(a) [Size of internal ROM[KBytes]
The internal ROM size to simulate is displayed (unit: Kbytes).
You cannot change the value of this property.

(b) [Size of internal RAM[KBytes]]
The internal RAM size to simulate is displayed (unit: Kbytes).
You cannot change the value of this property.

(2) [Endian]
You can configure endian in this category.

Figure 2.60 [Endian] Category [Simulator]

(a) [Endian of CPU]
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.

(3) [Clock]
You can configure clock in this category.

Figure 2.61 [Clock] Category [Simulator]

Clock	
System clock (ICLK) frequency[MHz]	50

(a) [System clock (ICLK) frequency [MHz]]
Specify the clock frequency for the CPU (unit: MHz).
You can specify a desired frequency value.
Enter an integer directly in the range 1 to 1,000.

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

(4) [Peripheral Function Simulation]
You can configure simulation of peripheral functions in this category.

Figure 2.62 [Peripheral Function Simulation] Category

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

(a) [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.

(b) [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.

2.3.5.2 [Debug Tool Settings] tab

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

(1) [Memory]
(2) [Access Memory While Running]
(3) [Register]
(4) [Trace]
(5) [Coverage]
(6) [Stream I/O]
(7) [Execution mode]
(8) [Instruction decode cache]

(1) [Memory]
You can configure the memory in this category.

Figure 2.63 [Memory] Category [Simulator]

Memory	
Memory mappings	**[17]**
[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

(a) [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. 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 for details on how to configure the parameters.

Figure 2.64 Opening the Memory Mapping Dialog Box

Memory	
Memory mappings	**[17]**
[0]	Internal RAM area
[1]	Non-map area

(2) [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 "2.11.1.4 Displaying and changing memory contents during program execution" for details on the real-time display update function.

Figure 2.65 [Access Memory While Running] Category [Simulator]

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

(a) [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).

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

(b) [Display update interval[ms]]
This property is displayed 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).
Note that if you've changed the specified value of the [Update display during the execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(3) [Register]
In this category, make settings related to PC display in the Status bar during program execution.

Figure 2.66 [Register] Category [Simulator]

Register	
PC display during the execution	Yes
Display update interval for PC[ms]	500

(a) [PC display during the execution]
This property specifies whether the PC value is displayed in the Status bar during program execution.
When you select [No], the Status bar under execution will show "Running."

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

(b) [Display update interval for PC[ms]]
This property is displayed only when you've selected [Yes] in the [PC display during the execution] property.
During program execution, specify a PC display updating interval in the Status bar in 100 ms units.
Enter an integer directly in the range 100 to 65500 (with fractions below 100 ms rounded up). (By default, [500] is specified.)
Note that if you've changed the specified value of the [PC display during the execution] property from [No] to [Yes], the previous set value is displayed in this property.

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

(4) [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.

(5) [Coverage]
You can configure the coverage function in this category.
See "2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]" for details on the coverage function and this category configuration.

(6) [Stream I/O]
You can configure the stream I/O for performing standard I/O or file I/O to/from the user program in this category.

Figure 2.67 [Stream I/O] Category

Stream I/O	
Select stream I/O mode	Simulator mode
Use stream I/O function	No
Stream I/O address	HEX 0

(a) [Select stream I/O mode]
In this property, specify an I/O mode from the drop-down list in which mode you want standard I/O to be executed.
To use files for the same low-level interface routines (assembly language part) as the emulator, specify [Emulator mode]. (By default, [Simulator mode] is specified.)

Remark File I/O is usable only when [Simulator mode] is specified. File I/O cannot be used when [Emulator mode] is specified.

Caution This property cannot be changed when a simulator is connected.

(b) [Use stream I/O function]
This property is displayed only when [Simulator mode] is specified in the [Select stream I/O mode] property.
Specify from the drop-down list whether or not to use the stream I/O function while simulator mode is in use.
Select [Yes] when using this function ([No] is selected by default).

(c) [Stream I/O address]
This property is displayed only when [Simulator mode] is specified in the [Select stream I/O mode] property.
Specify the position at which stream I/O starts while simulator mode is in use.
Directly enter the desired address ([0x00000000] is specified by default).

(7) [Execution mode]
You can configure the operation that is required in the event of simulation error or exception in this category.

Figure 2.68 [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

(a) [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.)

(b) [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).

(c) [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).

(d) [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).

(e) [Stop when floating-point exception is encountered][The microcontrollers with the FPU]
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 single-precision floating-point exception is encountered. Select [Yes] if you wish to stop at the occurrence of single-precision floating-point exception (default).

(f) [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).

(g) [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).

(h) [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).

(8) [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 2.69 [Instruction decode cache] Category

Instruction Decode Cache	
Cache the results of decoding instructions and accelerate simulation	No

(a) [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.

2.3.5.3 [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.

2.3.5.4 [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.

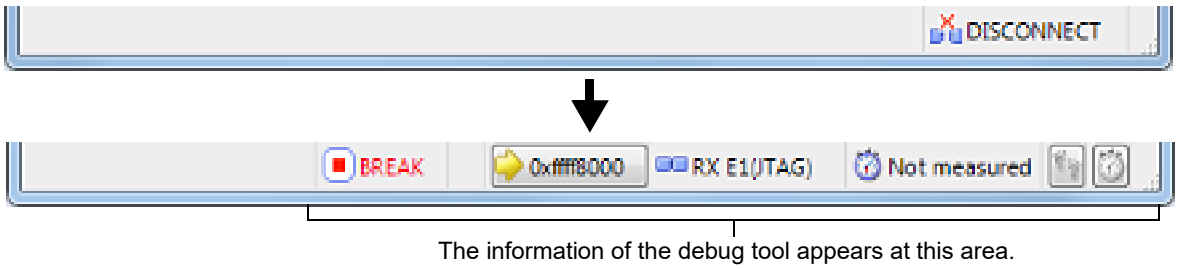
2.4.1 Connect the debug tool to CS+

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 2.70 Status Bar 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.

(1) Display the Version Information [E1] [E20] [EZ Emulator]
After successful connection to the debug tool, the version information such as the emulator firmware version and the emulator information such as supplied voltage are displayed on the Output panel.

Figure 2.71 Output Panel after Connection [E1]

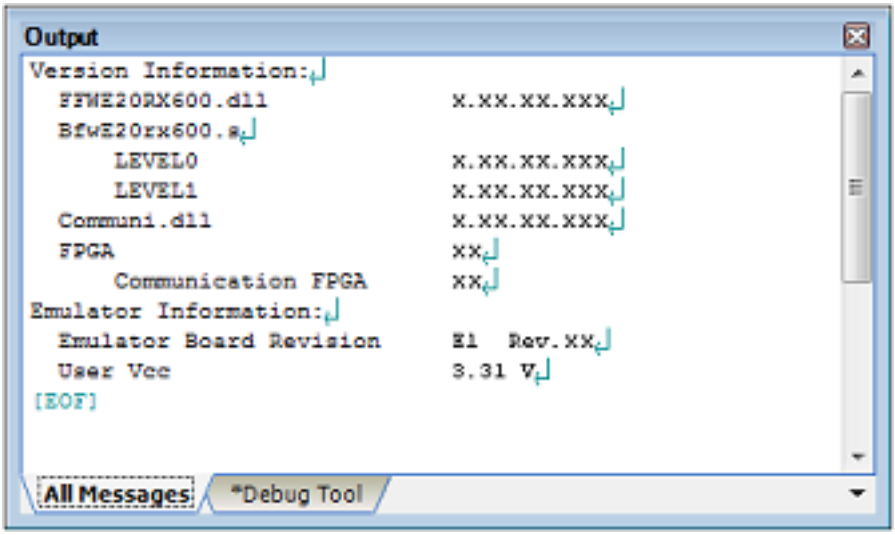


Figure 2.72 Output Panel after Connection [E2 Lite]

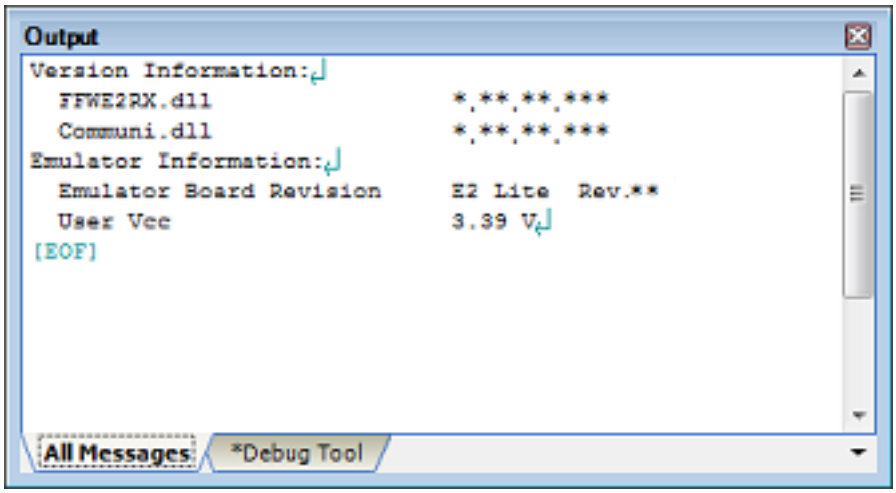


Figure 2.73 Output Panel after Connection [E2]

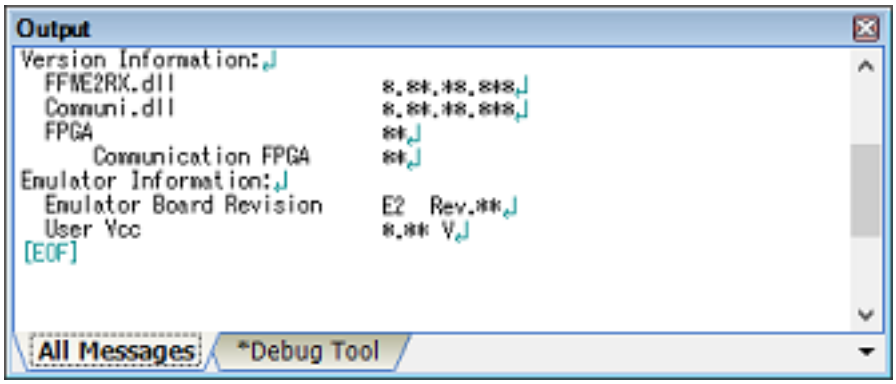


Figure 2.74 Output Panel after Connection [E20]

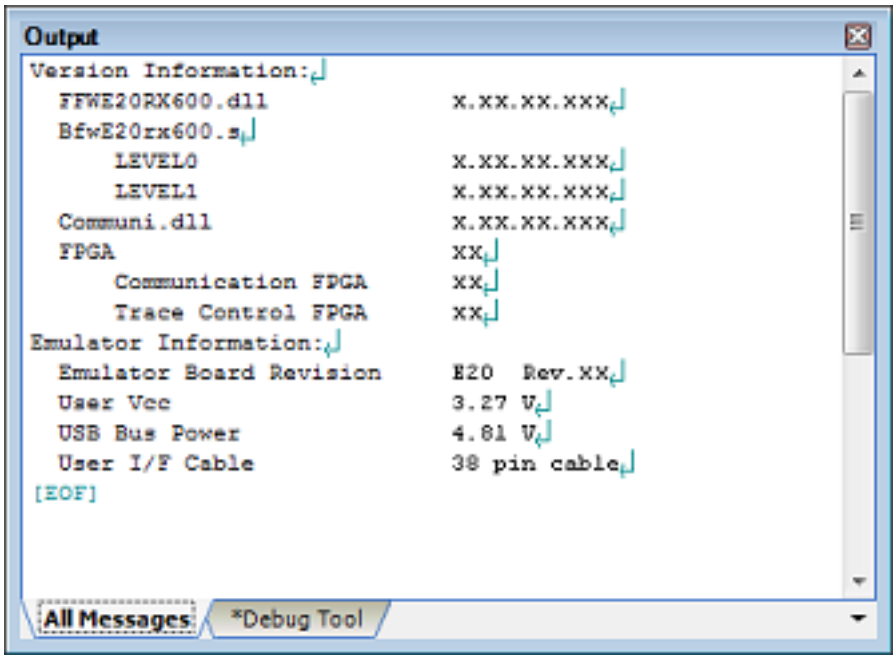
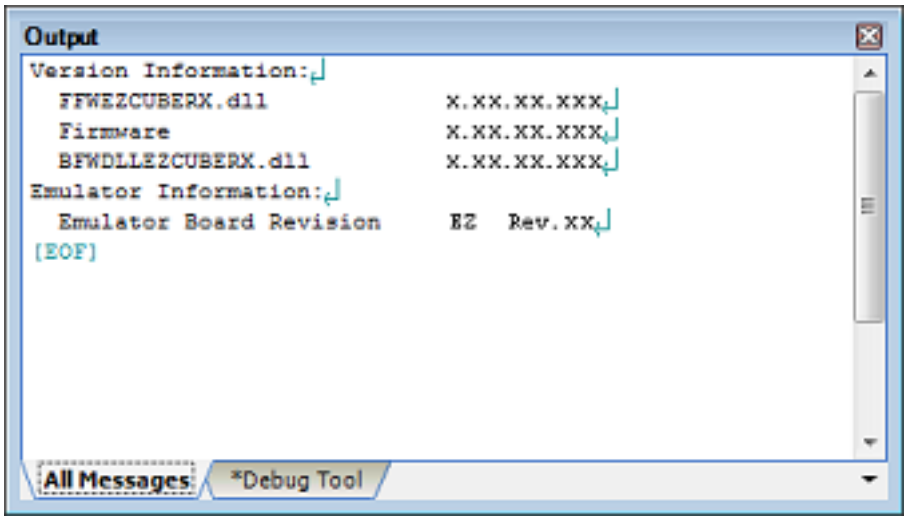


Figure 2.75 Output Panel after Connection [EZ Emulator]

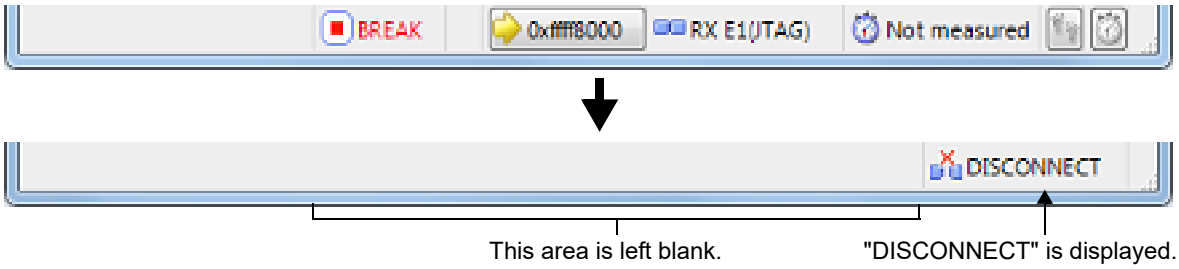


Caution [E1] [E20]
When the emulator firmware should be updated, it is automatically updated. To ensure this update, do not disconnect the USB or power supply until connection is established.

2.4.2 Disconnect the debug tool from CS+

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 2.76 Status Bar Indicating the Disconnection from the Debug Tool



Caution 1. The debug tool cannot be disconnected from CS+ while the program is running. If you wish to disconnect the debug tool, stop the program in advance.

Caution 2. [E1] [E20] [EZ Emulator]
If you are using the E1, E20, or EZ Emulator, select the above or either of the following ways to disconnect the debug tool from CS+.
(1) Select [Disconnect from Debug Tool] from the [Debug] menu.
(2) Enter “debugger.Disconnect()” in the Python Console panel.
If you terminate the debugger, you will not be able to start up the debugger correctly next time. In such a case, turn the power off and turn it on again (i.e. turn off the power switch on the E20 and turn it on or disconnect the USB interface cable from the E1 or EZ Emulator and re-connect it). When using EZ Emulator, also restart CS+.

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 CS+ using hot plug-in [E1(JTAG)] [E20(JTAG)]

With hot plug-in function, you can connect the debug tool to the CS+ 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.

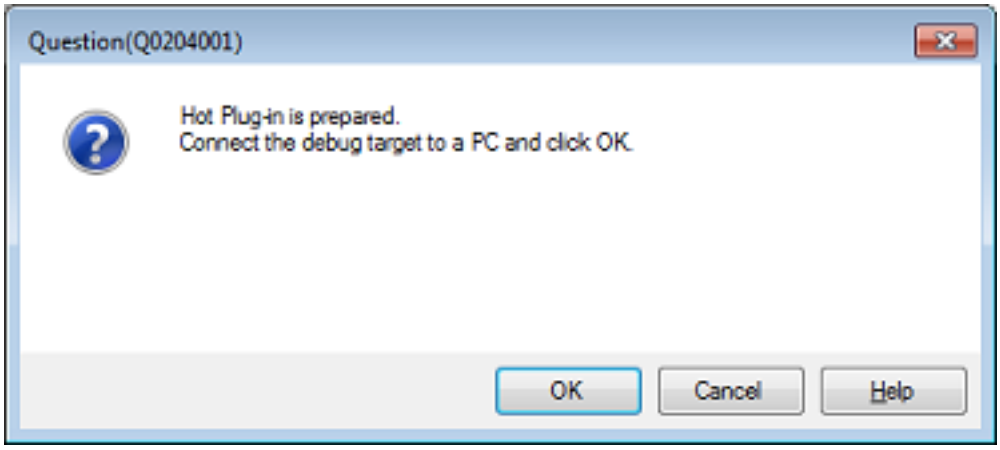
(2) Specify the debug tool
In the active project, specify the debug tool which supports hot-plug in connection ([E1(JTAG)]/[E20(JTAG)]).

Remark **[E1(Serial)] [E20(Serial)] [EZ Emulator]**
Hot plug-in connection is not supported.

(3) Connect the debug tool to CS+ 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.

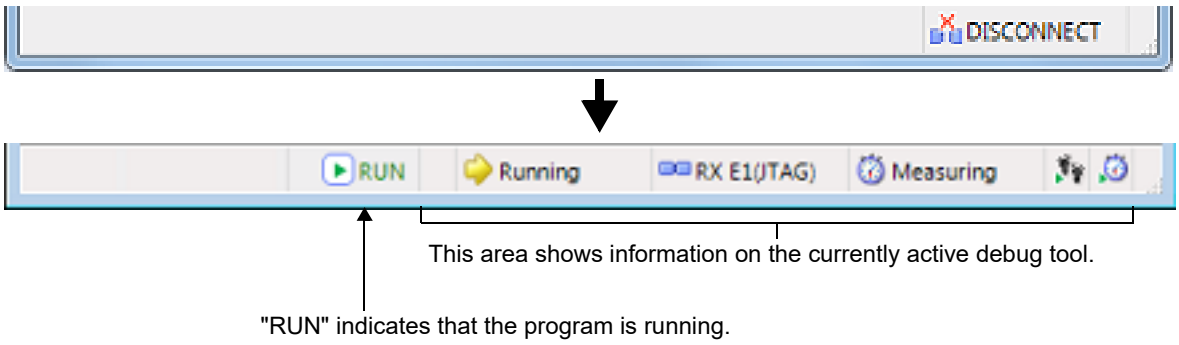
Figure 2.77 Message Indicating that Hot Plug-in Connection Is Ready to be Started



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 2.78 Status Bar Indicating the Successful Hot plug-in Connection to the Debug Tool



Caution 1. The trace function will not be available until the program breaks following the completion of hot plug-in connection.

Caution 2. The real-time 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.

Caution 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.

Caution 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).

Caution 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).

Caution 6. [RX630, RX631, RX63N, RX63T, and RX634 Groups]
Confirm that the endian settings of the endian select registers (MDEB, MDES) written on the microcontroller match the endian information of the project. See the hardware manual of the microcontroller for the details of endian select registers.
Hot plug-ins cannot be used while a flash rewrite program is running.

2.5 Download and Upload

This section describes how to download the program to be debugged (e.g., load module files) into CS+, and how to upload the memory content under debug from CS+ 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 2.79 [Download] Category [E1(JTAG)] [E1(Serial)] [E20(JTAG)] [E20(Serial)] [EZ Emulator]

Download	
Download files	**[1]**
[0]	DefaultBuild\sample.abs
File	DefaultBuild\sample.abs
File type	Load module file
Download object	Yes
Download symbol information	Yes
Specify the PIC/PID offset	No
Generate the information for input completion	Yes
CPU Reset after download	Yes
Erase flash ROM before download	Yes
Erase data flash ROM before download	No
Automatic change method of event setting position	Suspend event

Figure 2.80 [Download] Category [Simulator]

Download	
Download files	**[1]**
[0]	DefaultBuild\sample.abs
File	DefaultBuild\sample.abs
File type	Load module file
Download object	Yes
Download symbol information	Yes
Specify the PIC/PID offset	No
Generate the information for input completion	Yes
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 "[ ]" indicates 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 CS+), a debug-dedicated project needs to be created.
If you use a debug-dedicated 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-dedicated projects," see the separate edition, "CS+ Integrated Development Environment User's Manual: Project Operation".

(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] [E1] [E20] [EZ Emulator]
Specify from the drop-down list whether or not to erase the flash ROM (program ROM) before a download is executed.
To erase the flash ROM, select [Yes].

(d) [Erase data flash ROM before download] [E1] [E20] [EZ Emulator]
Specify from the drop-down list whether or not to erase the data flash ROM before a download is executed.
To erase the data flash ROM, select [Yes]. (By default, [No] is selected).

(e) [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 2.81 [Debug information] Category

Debug Information	
Execute to the specified symbol after CPU Reset	Yes
Specified symbol	_main
Specify the debugged overlay section	No
The upper limit size of the memory usage [MBytes]	500

(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.

(e) [The upper limit size of the memory usage[MBytes]]
Specify the maximum size [Mbytes] of memory to be used to read debug information.
Directly enter an integer between 100 and 1000 ([500] is specified by default).
When an insufficient memory error occurs, specifying a smaller value for this maximum size may reduce the occurrence of the error. However, a smaller maximum size may degrade the response speed of the debug tool.

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 CS+ 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.

(4) Canceling a download
To cancel a download, click on the [Cancel] button on the Progress Status dialog box, which displays the progress of downloading, or press the [Esc] key.

When 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.
CS+ permits files in the following formats to be downloaded.

Table 2.2 Downloadable File Formats

File Format	Extension	Remark
Load module format	abs	
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 format - (S0, S1, S9-16 bits) - (S0, S2, S8-24 bits) - (S0, S3, S7-32 bits)	mot	
Binary data format	bin	

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 2.82 Opening the Download Files Dialog Box

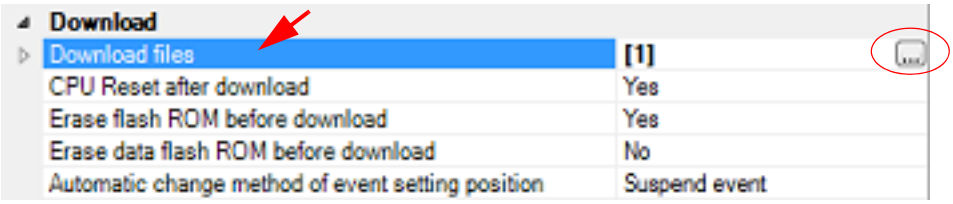
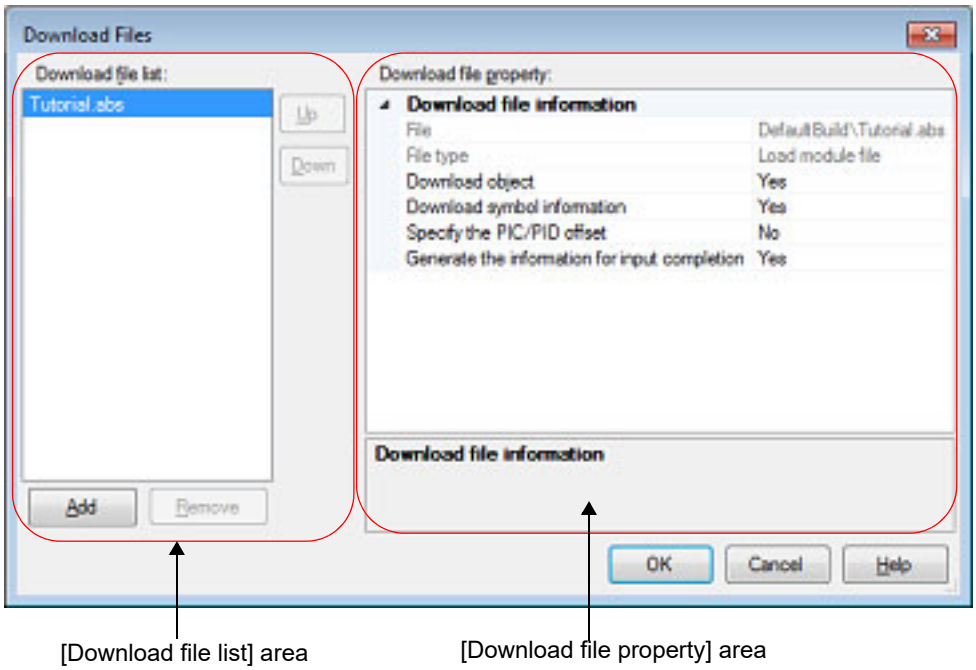


Figure 2.83 An Applied Method of Download (Download Files Dialog Box)



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

2.5.2.1 Changing the download conditions for load module files
2.5.2.2 Adding a download file (*.hex, *.mot, or *.bin)
2.5.2.3 Downloading multiple load module files
2.5.2.4 Performing source-level debug with hex format, Motorola S format, or binary data format files
2.5.2.5 Downloading files to external flash memory [E1] [E20]

2.5.2.1 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.

(1) Selection of a load module file
In the [Download file list] area, select the load module file to download.

(2) 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	

Generate the information for input completion	This item is displayed only when the files to be downloaded are in the load-module format. Specify whether or not to generate information for use by the auto-complete feature while symbol information is being downloaded from a specified file. If you use the symbol name completion function in "Watch panel", "Memory panel", select is "Yes". Download time is long in order to generate this information during the download.		
	Default	Yes	
	How to change	Select from the drop-down list.	
	Specifiable value	Yes	Generate information for use by the auto-complete feature
		No	Does not generate information for use by the auto-complete feature

Note 1. Unless symbol information is downloaded, source-level debugging cannot be performed.

Note 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.

(3) Clicking the [OK] button
Settings you've made in this dialog box are enabled, with the download conditions changed.

2.5.2.2 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.

(1) 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.

(2) 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.

(3) 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.

(4) 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).

2.5.2.3 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.

Remark When multiple load module files are downloaded, the load module names are displayed as the detailed information on the Events panel.

(1) 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.

(2) 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

Note 1. Unless symbol information is downloaded, source-level debugging cannot be performed.

Note 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.).

(3) 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.

(4) 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).

2.5.2.4 Performing source-level debug with hex format, Motorola S format, or binary data format files

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.

(1) 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.

(2) 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.

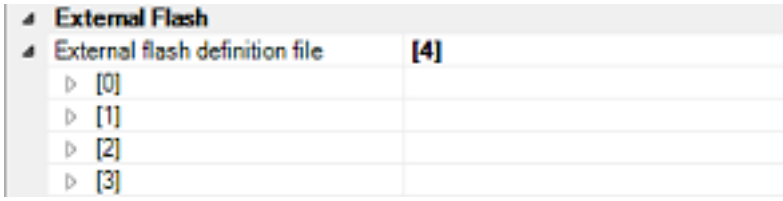
(3) Clicking the [OK] button
Settings you've made in this dialog box are enabled, with the specified load module file added as a download file
(Only the symbol information included in the load module file becomes the subject to download.).

2.5.2.5 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.

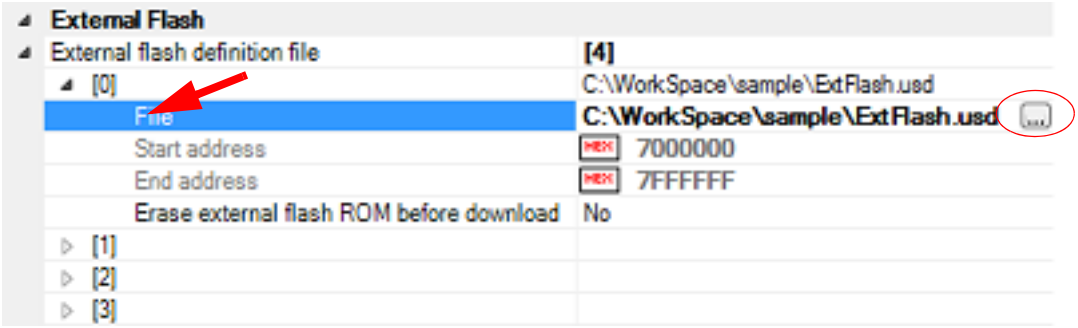
(1) Setting up the Property panel
In the [External Flash] [E1] [E20] category on the Property panel's [Debug Tool Settings] tab, register an external flash definition file (USD file).
External flash memory is recognized by registering a USD file in the [External Flash] [E1] [E20] category. Up to four USD files can be registered.
By clicking the "+" mark of an external flash definition file, the definition file name, start and end addresses, and whether to erase the external flash ROM before downloading are displayed for each index ([1] to [4]).

Figure 2.84 [External Flash] **Category**



(a) [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.
After a USD file is registered, the [Start address], [End address], and [Erase external flash ROM before download] sub-properties are added to the displayed information.
For the [Start address] and [End address] properties, the external flash memory area defined in the registered USD file is displayed (this information cannot be modified).
For the [Erase external flash ROM before download] property, select [Yes] or [No] from the drop-down list.
To erase the external flash ROM before downloading, select [Yes]. ([No] is selected by default.)
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 Definition Editor.

Figure 2.85 Opening the External flash memory Dialog Box



Caution 1. 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.

Caution 2. When the address ranges of multiple registered USD files overlap each other, connection with the debug tool cannot be established.

Caution 3. For downloading in external flash memory, internal RAM or CS-area RAM can be used as a work area. Note that SDRAM areas (SDCS) are not usable.

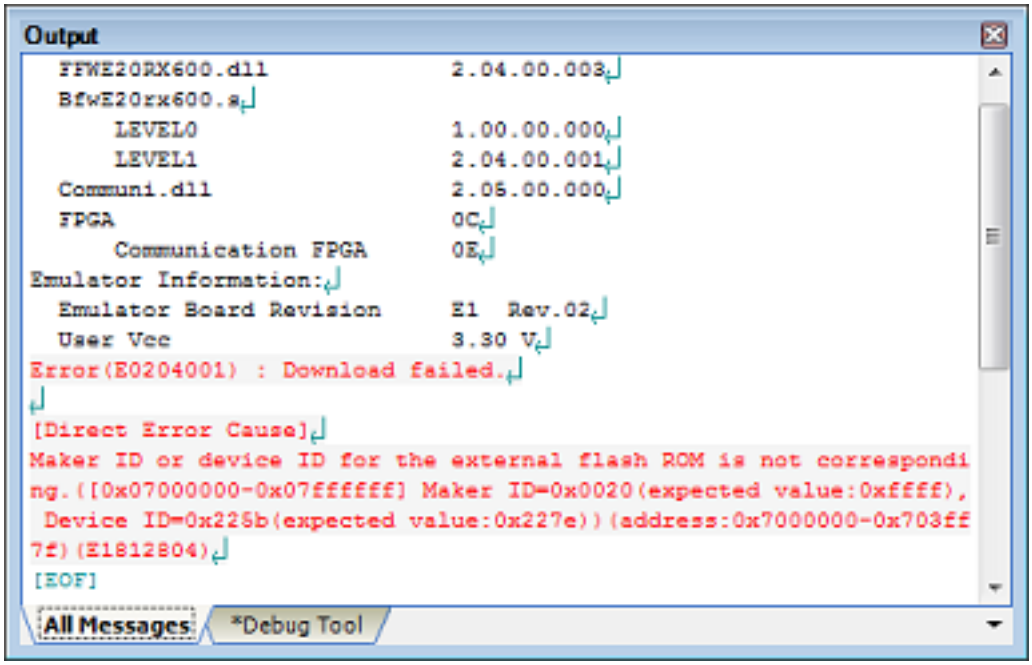
Caution 4. To allocate a device to multiple CS areas, set [Erase external flash ROM before download] to [No] in the Property panel.

Caution 5. The work RAM area can also be used by the user program because the emulator saves and restores data in this area. Note, however, that the work RAM area is not specifiable as:

- The destination or origin of a DMA or DTC transfer
- An address where a DTC vector table or transfer information is to be allocated
- The interrupt vector for a DMAC or DTC activation source

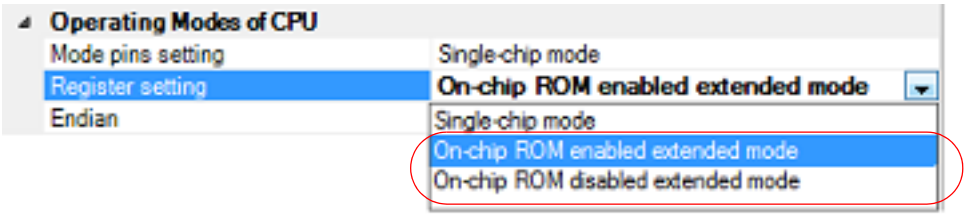
Remark If the manufacturer ID or device ID does not match, the read value and expected value are displayed in the Output panel.

Figure 2.86 Error Messages in the Output Panel



(2) 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] [EZ Emulator] 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 pull down list, as shown below.

Figure 2.87 Specifying the CPU Operation Mode



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

Caution 1. External flash memory cannot be rewritten on the Memory panel.

Caution 2. When downloading to the external flash memory, if you are using external RAM as working RAM, set it to the same endian as the CPU.

Caution 3. Only flash memory with 4096 or fewer sectors can be registered. If flash memory with more sectors is connected, programming cannot be guaranteed.

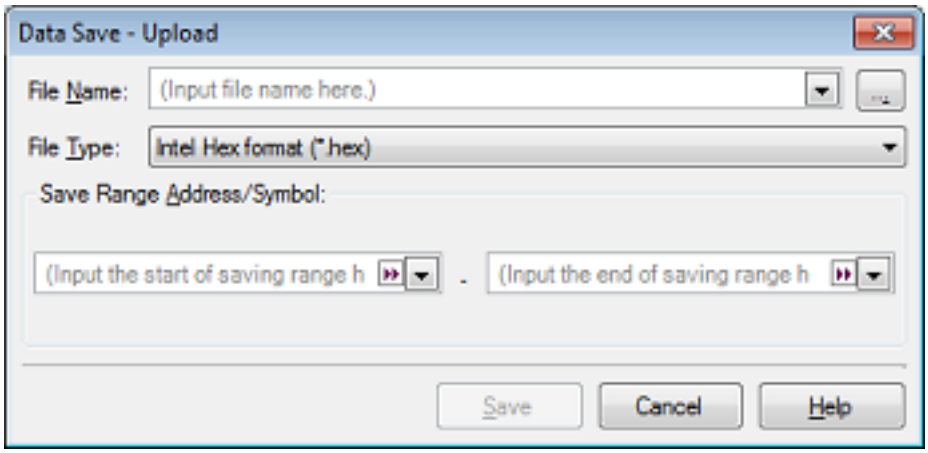
Caution 4. To use a flash memory device that does not support the use of the Lock command, specify that the lock bit be cleared when you generate USD files. This specification helps you omit an unnecessary process to confirm the lock bit status.
Likewise set [Erase external flash ROM before download] to [No] in the Property panel.
* The lock command refers to a command to be executed in the following command sequence.
command sequence :

	1st cycle	2nd cycle	3rd cycle	4th cycle
Address	555H	2AAH	555H	Block address
Data	AAH	55H	48H	X1H / X0H

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 2.88 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:

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 "2.21.2 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 disassembled 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 "2.5.2.4 Performing source-level debug with hex format, Motorola S format, or binary data format files").

2.6.1 Displaying source files

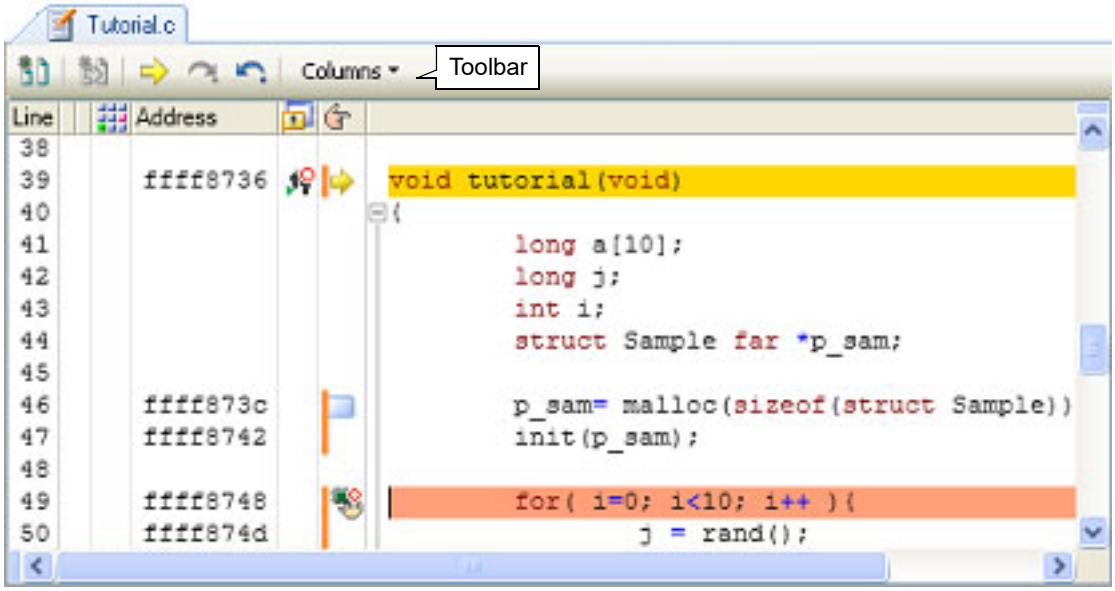
Use the Editor panel shown below to display the source file.

The Editor panel opens automatically after a load module file is successfully downloaded, with the source text at the position you specified (see "2.5.1 Execute downloading") displayed on the panel.

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 "CS+ Integrated Development Environment User's Manual: Editor".

Figure 2.89 Displaying the Source File (Editor Panel)



2.6.2 Displaying the disassembled result

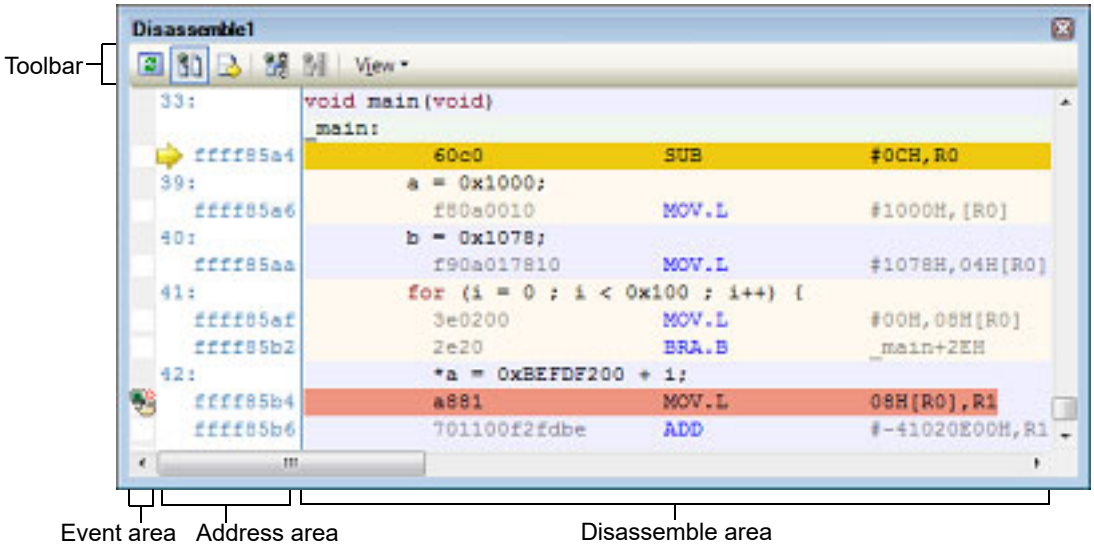
To display the disassembled result (disassembled text) of a downloaded program (memory content), use the Disassemble panel shown below.

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 2.90 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.

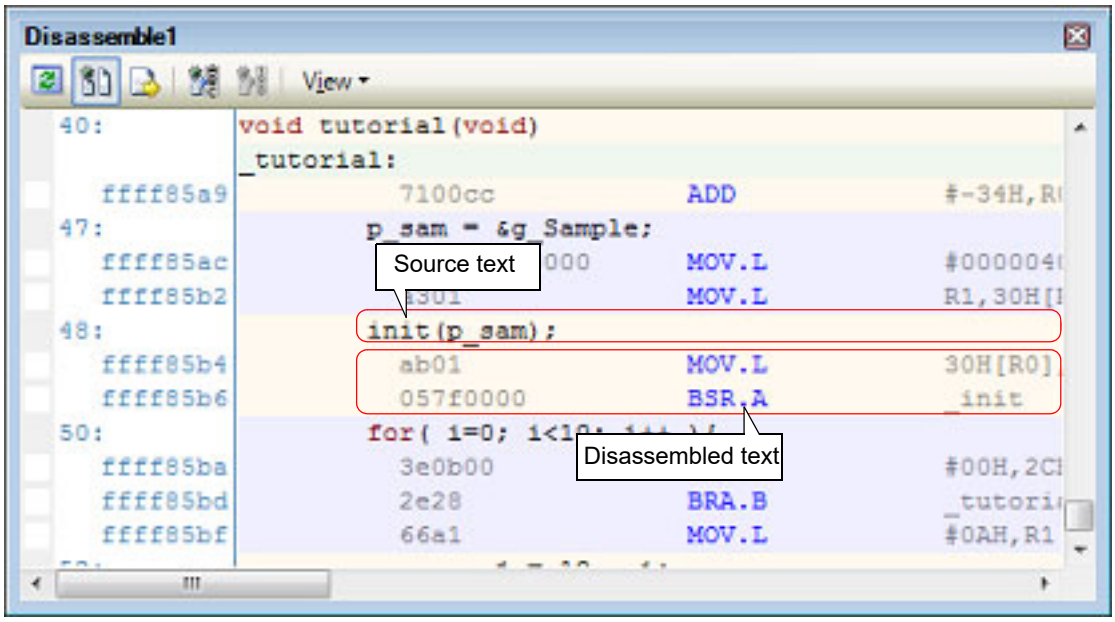
2.6.2.1 Changing the display mode
2.6.2.2 Changing the display form
2.6.2.3 Moving to a specified address
2.6.2.4 Moving to a symbol definition part
2.6.2.5 Saving the displayed contents of disassembled results

2.6.2.1 Changing the display mode

You can change the display mode of the Disassemble panel by clicking the button (toggle) on the toolbar.

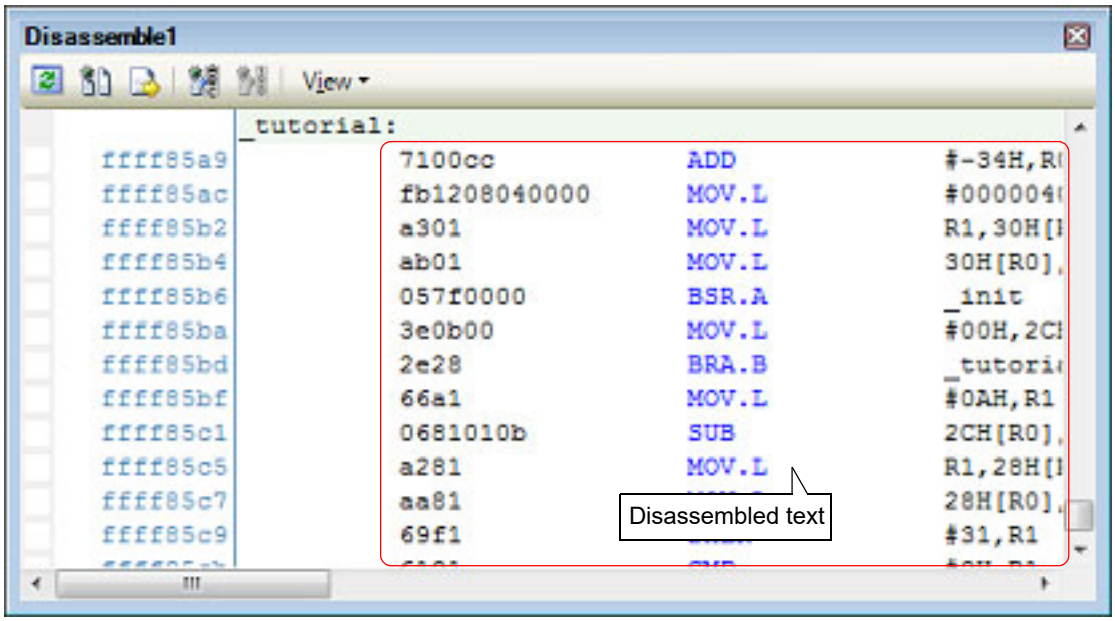
- Mixed display mode
  In this display mode (default), the disassembled text is displayed combined with the source text.

Figure 2.91 Mixed Display Mode (Disassemble Panel)



- Disassemble display mode
  In this display mode, the source text is hidden and only the disassembled text is displayed.

Figure 2.92 Disassemble Display Mode (Disassemble Panel)



2.6.2.2 Changing the display form

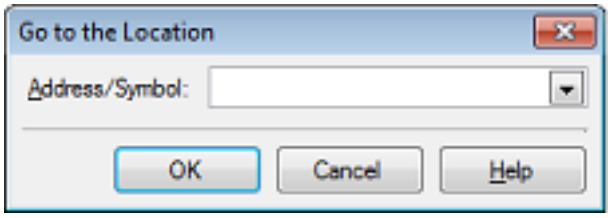
The form in which disassembled results are displayed can be changed freely by using the toolbar buttons shown below.

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.

2.6.2.3 Moving to a specified address

To move to a specified address in the disassembled text, select [Go To...] from the 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 2.93 Move to the Specified Address in Disassembled Result (Go to the Location Dialog Box)



(1) Specifying [Address/Symbol]
Specify an address to which you want to move the caret.
Enter an address expression directly in the text box (specifiable in up to 1024 characters) 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 "2.21.2 Symbol name completion function").

(2) Clicking the [OK] button
The caret moves to the specified address.

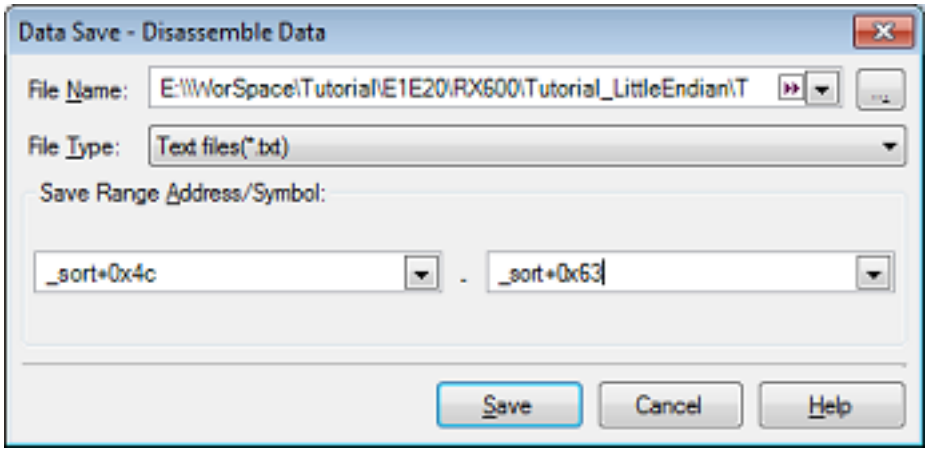
2.6.2.4 Moving to a symbol definition part

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.

2.6.2.5 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, CS+ gets latest information from the debug tool and saves it in the form in which data are displayed on this panel.
Choose [Save Disassemble Data As...] from the [File] menu, and the Data Save dialog box shown below is opened. (At this time, if you perform this operation while a range is selected on panel, 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 2.94 Saving Disassembled Result (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 using 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) (*.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.

(3) 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 "2.21.2 Symbol name completion function").

(4) Clicking the [Save] button
Disassembled data is saved in specified format to a specified file.

Figure 2.95 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
   :        :       :                 :
```

Remark 1. If panel contents are saved over an existing file by selecting [Save Disassemble Data] on [File] menu, the respective Disassemble panels (Disassemble1-4) are handled individually.
Also, as to the save range, the previously specified address range is applied when data is saved.

Remark 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

CS+ 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] check box in the [General - Build/Debug] category of the Option dialog box.

Remark 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.

Remark 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).

Remark 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.

2.6.4.1 Editing instructions
2.6.4.2 Editing instruction code

2.6.4.1 Editing instructions

To edit instructions, follow the procedure described below.

(1) 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] from the context menu, and the subject you're going to edit is placed into the edit mode.

(2) Editing instructions
Edit the character string of the instruction directly from the keyboard.

(3) 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 compensated for with nop instructions as in the example shown below.

Example 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

Example 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

2.6.4.2 Editing instruction code

To edit instruction code, follow the procedure described below.

(1) 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] from the context menu, and the subject you're going to edit is placed into the edit mode.

(2) Editing instruction code
Edit the character string of instruction code directly from the keyboard.

(3) 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.

Remark 1. For details on the PIC/PID function, see "CC-RX Compiler User's Manual".

Remark 2. For setting of the PIC/PID function by the build tool, see "CS+ Integrated Development Environment User's Manual: CC-RX Build Tool Operation".

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 2.96 [Download] Category

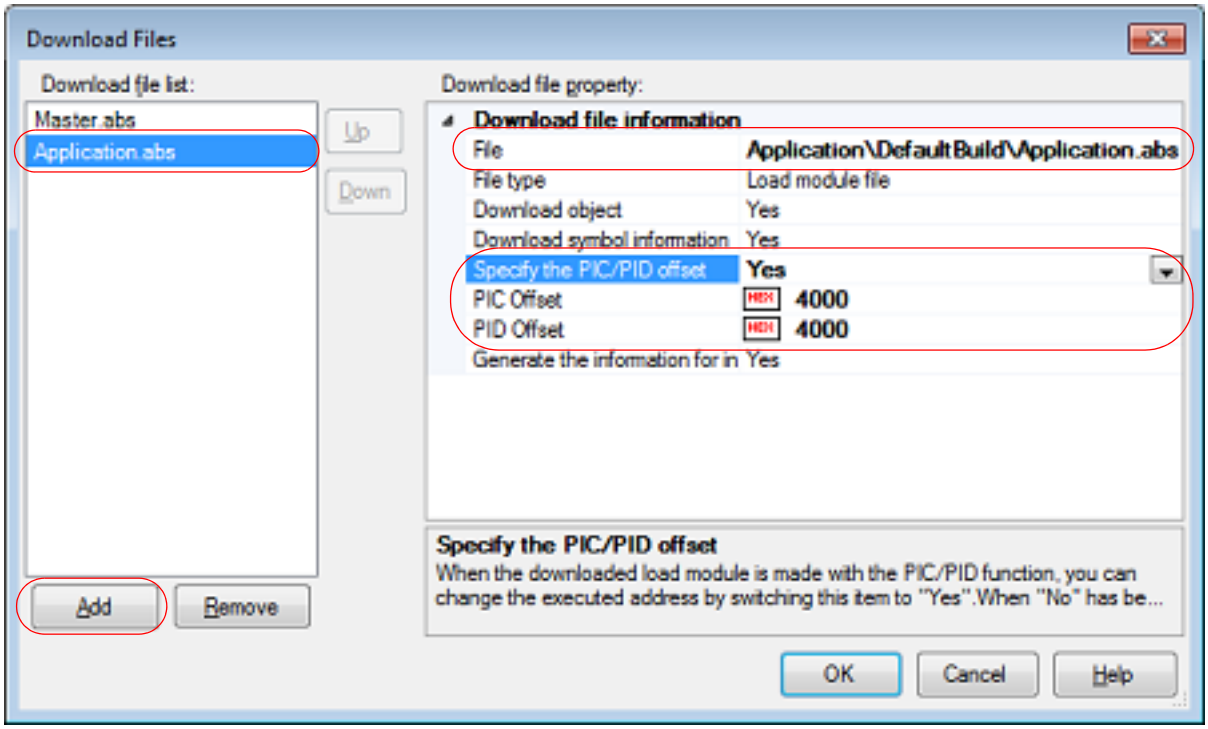
- [Download files]
  Click on the [...] button to open the Download Files dialog box.

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 2.97 Adding a Download File and Changing Conditions for Downloading (Download Files Dialog Box)

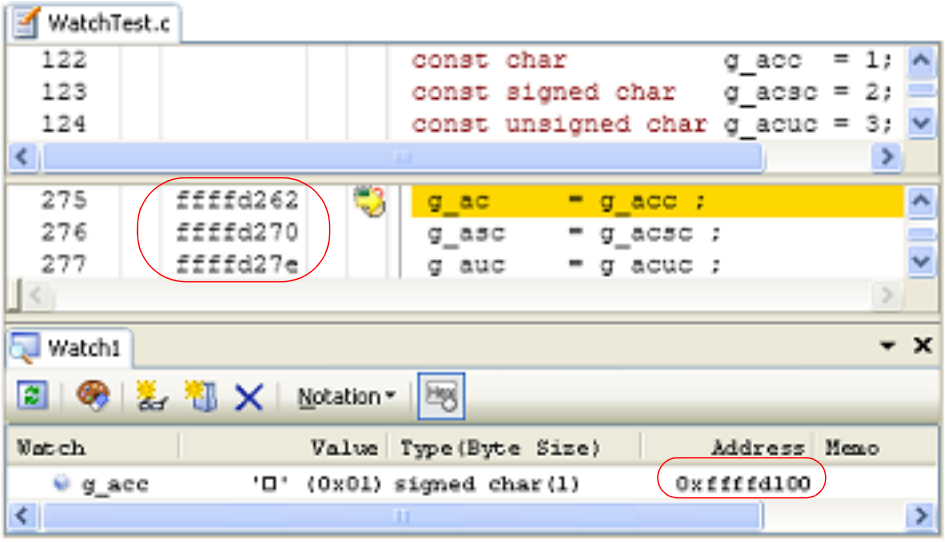


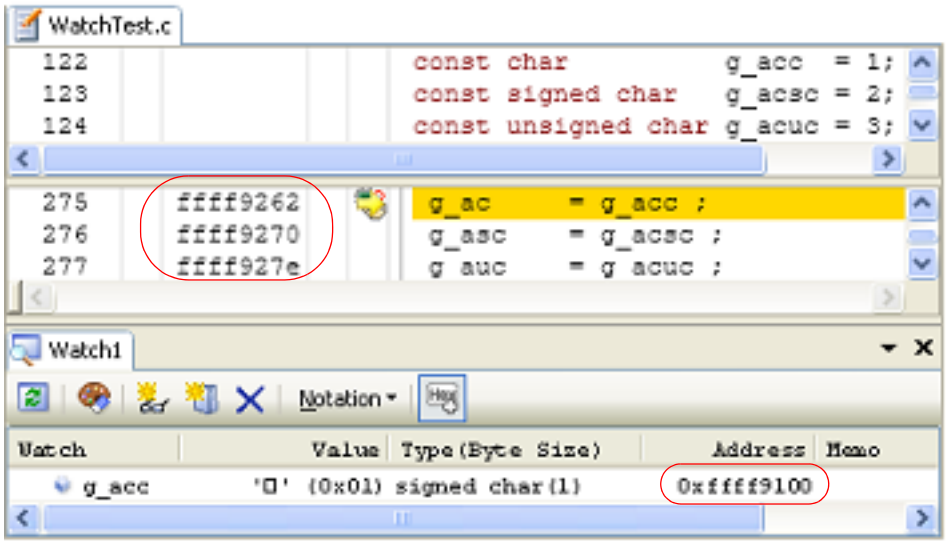
When the load module is downloaded after the values of [PIC Offset] and [PID Offset] have been changed, the allocation of addresses for the P section and external or static variables is changed as shown in the figure below.

The figure shows examples of downloading a load module; "4000" and "0" are specified for [PIC Offset], [PIROD Offset], and [PID Offset] in the Download Files dialog box in the figures above and below, respectively.

In the figure above, "0x4000" is added to the base address.

Figure 2.98 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.

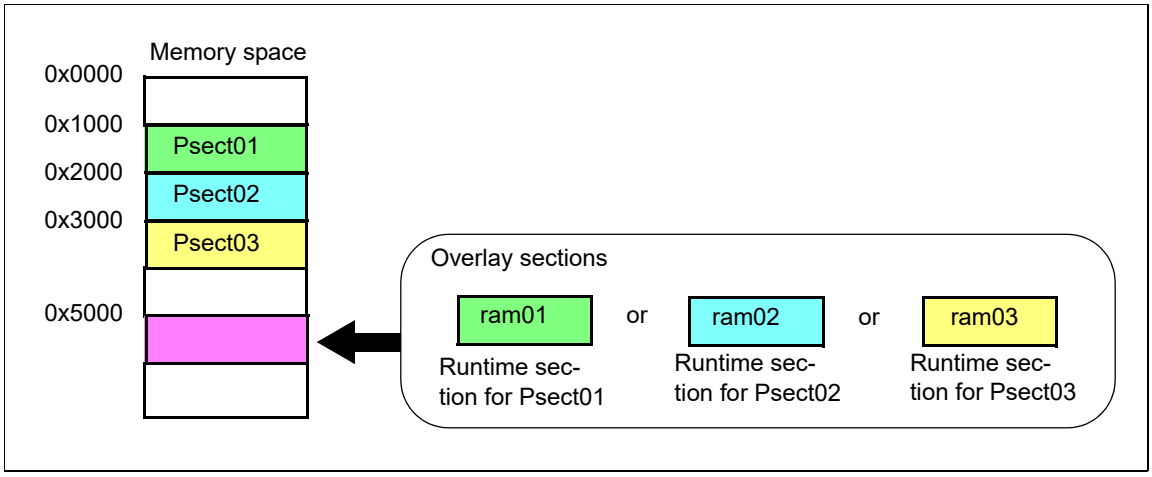
Example Defining overlay sections by Optlnk

```
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, see "CS+ Integrated Development Environment User's Manual: CC-RX Build Tool Operation".

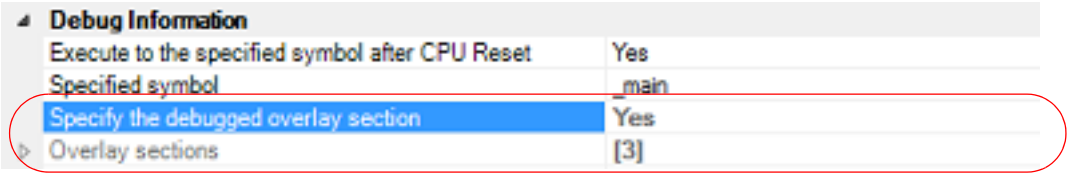
Figure 2.99 Allocation of Overlay Sections



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 2.100 [Debug Information] Category

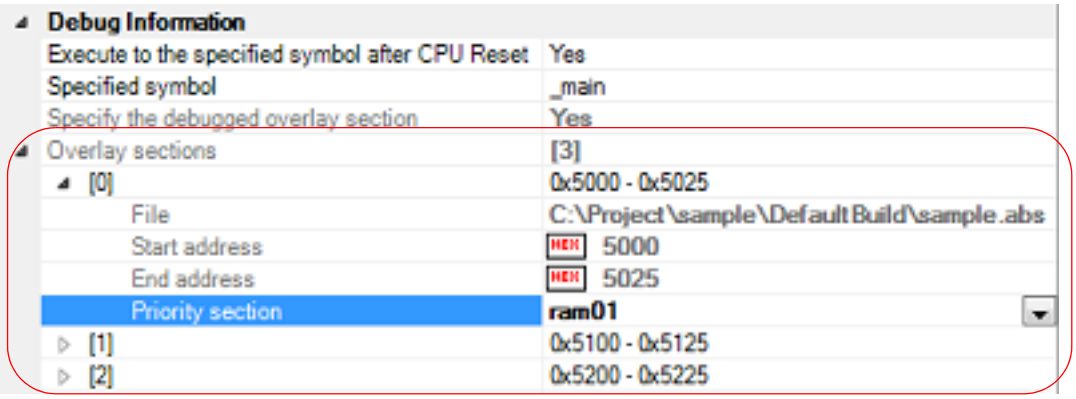


(1) [Specify the debugged overlay section]
"Yes" is displayed when overlay sections exist in the load module (this cannot be changed).

(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 2.101 [Overlay sections] Property



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

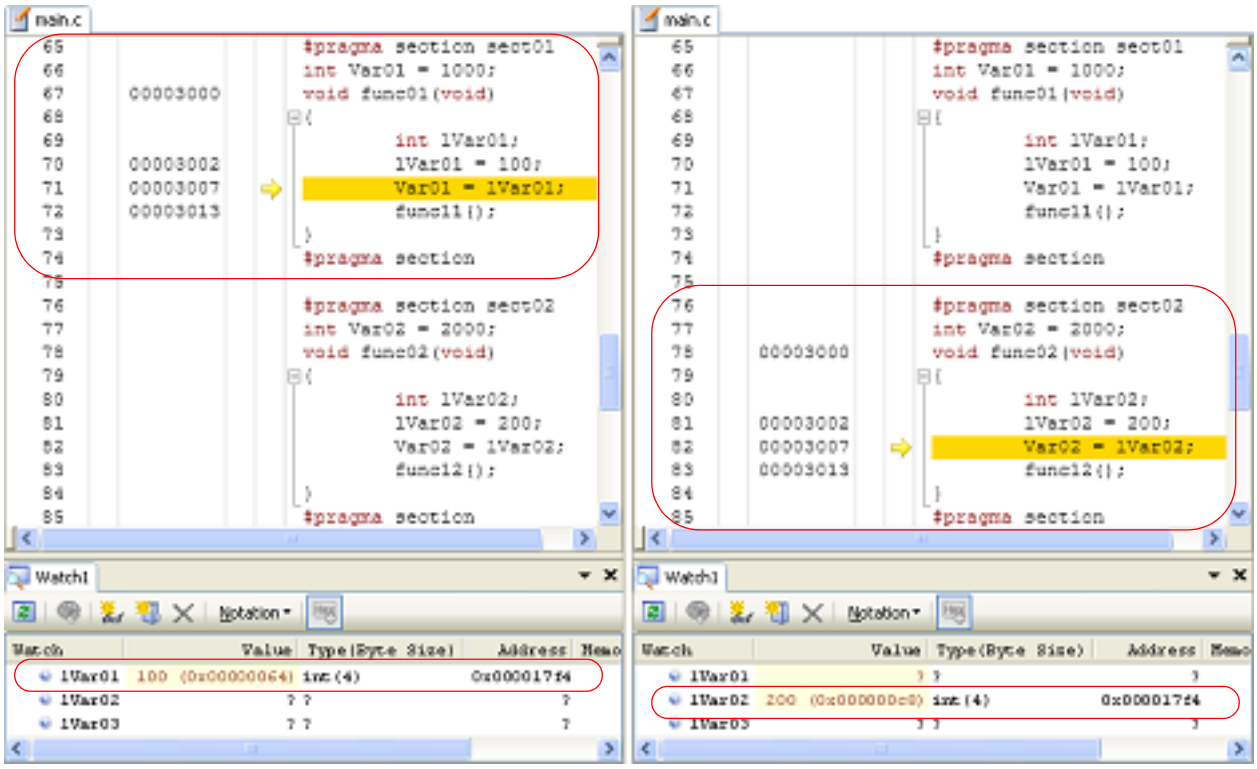
- [File]
  Name of the load module file in which the selected address group has been downloaded (this cannot be changed).
- [Start address]
  First address of the selected address group (this cannot be changed).
- [End address]
  Last address of the selected address group (this cannot be changed).
- [Priority section]
  List of sections defined in the selected address group. You can choose a section to be debugged (priority section) from this list.

Caution 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.

Caution 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 2.99 Allocation of Overlay Sections.

Figure 2.102 Changing the Selection for [Priority section] from "*ram01*" to "*ram02*"



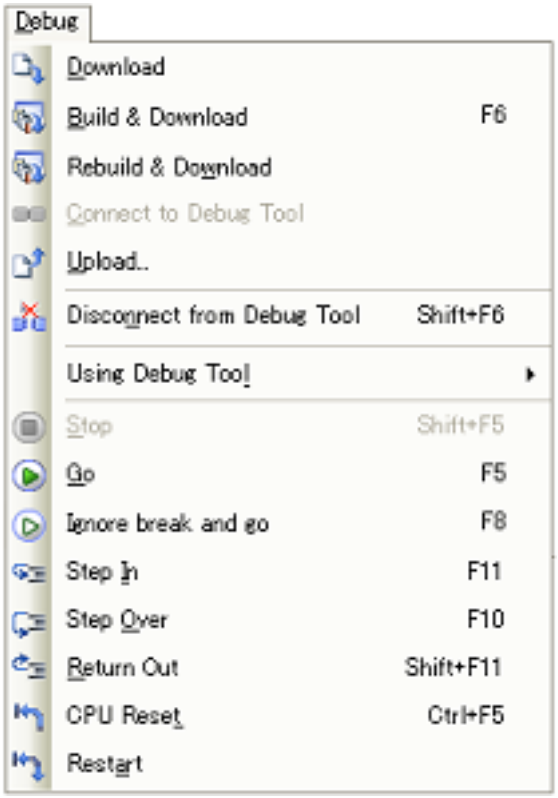
2.9 Execute Programs

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 2.103 Debug Toolbar

Figure 2.104 [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 CS+ 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.

2.9.2.1 Execute after resetting microcontroller (CPU)
2.9.2.2 Execute from the current address
2.9.2.3 Execute after changing PC value

Remark 1. 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).

Remark 2. The PC value can be displayed in the status bar at any update interval during program execution. (See "(e) Current PC position".)

2.9.2.1 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 "2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20] [EZ Emulator]" and "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.

2.9.2.2 Execute from the current address

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

(1) 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] [EZ Emulator]" and "2.10.4 Stop the program with the access to variables/I/O registers").
- Other break factors have occurred.

(2) 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.

(3) 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.

2.9.2.3 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 "2.9.2.2 Execute from the current address".

Caution If a program that has stopped at a software or hardware (pre-execution) breakpoint is restarted from the same address, the realtime capability is lost because stepped execution of the instruction at that address takes place before the program is restarted. Any events that are triggered by the stepped execution are cleared next time the program is run. Even if a trace or time-measurement start event is triggered by the stepped execution, tracing or time measurement does not start until the same event is triggered again. Likewise any break or trace start events that are triggered by the stepped execution are not effective as AND or sequential conditions.

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 by the setting of the Editor panel as follows:

- When the button on the toolbar is invalid (default):
  Step execution is conducted in source level.
  Note, however, that when the focus is in the Disassemble panel or the line information does not exist in the address specified by the current PC value, the step execution is conducted in instruction level.
- When the button on the toolbar is valid:
  Step execution is conducted in instruction level.

Remark The button is only enabled if the mixed display mode is selected on the Editor panel.

Step execution is divided into the following types:

2.9.3.1 Step into the function (Step In execution)
2.9.3.2 Step over the function (Step Over execution)
2.9.3.3 Execute until return is completed (Return Out execution)

Caution 1. Break points, break events, and Printf events that have been set do not occur during step execution.

Caution 2. [Simulator]
You may jump to an interrupt handler during step execution.

Caution 3. During source-level stepping, the debugger may appear to be executing instructions that are not supposed to be executed.
However, the reason for these problems is a difference between the debugging information generated by the compiler and the actual code. The result of executing the code generated by the compiler is correct.

Example In the program code shown below, it seems that the position indicated by the current PC might be moved to position (*1) after the execution of (*2), although (*1) is never actually executed in the generated code.

```
void main(void);
int x, y, z1, z2, z3;
void func(int i)
{
    if (i == 0) {
        ++x; // <-(*1)
        ++z1;++z2;++z3;
    } else {
        ++y; // <-(*2)
        ++z1;++z2;++z3;
    }
}
int one = 1;
void main(void)
{
  while (1)
  {
    func(one);
  }
}
```

Note that this caution may be eliminated by setting the [Optimization level] property to [0 (-optimize=0)] in the [Optimization] category on the [Compile Options] tab in the Property panel of the build tool.

2.9.3.1 Step into the 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.

Caution 1. Step In execution cannot be performed for a function that has no debug information.

Caution 2. If Step in execution is performed for the longjmp function, program execution may not complete and may wait for a time-out.

Caution 3. [E1] [E20] [EZ Emulator]
No interrupts are accepted during stepped execution. In JTAG communication, resetting is also prohibited during stepped execution.

2.9.3.2 Step over the 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 1. If Step over execution is performed for the longjmp function, execution processing may not complete and may wait for a time-out.

Caution 2. [E1] [E20] [EZ Emulator]
While stepping over a function or subroutine call, interrupts and resetting are allowed.

2.9.3.3 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 calling 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 calling function without step executing the remaining instructions inside the function. This instruction can be performed by clicking the button on the debug toolbar.

Caution 1. If a program is returned out in the main function, it will break inside the start-up routine.

Caution 2. If a program is returned out in a function that has called the longjmp function, break may not occur.

Caution 3. Executing Return Out from the recursive function may cause the program to run in a free-run mode.

Caution 4. Executing Return Out from a function that uses a jump instruction (i.e. not an RTS (return from subroutine) instruction) to return to the calling function is not possible.

Caution 5. [E1] [E20] [EZ Emulator]
While returning out in a function or subroutine, interrupts and resetting are allowed.

Caution 6. Executing a return-out command from a recursive function may lead to execution stopping at the address of the return instruction in the called function instead of the correct line in the calling function.

2.9.4 Execute a specified routine [E1] [E20] [EZ Emulator]

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] [EZ Emulator] 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 [Routine to run immediately before execution starts] 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 the user program stops] property, and enter the start address in the [Routine to run immediately after execution stops] property that is displayed. The specified routine will be executed immediately after the stop of the program.

If [Yes] is selected either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property, the [Work RAM start address for executing a specified routine] property is displayed. The amount of memory indicated by the [Work RAM size [bytes] for executing a specified routine] this property beginning with this address is to be used by the debugger firmware as a work RAM area for the facility to execute a specified routine.

When you wish to use the facility to execute a specified routine, select this work RAM area.

Figure 2.105 [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	
Work RAM start address for executing a specified routine	HEX 1DD0
Work RAM size [bytes] for executing a specified routine	560

Caution 1. In the [Routine to run immediately before execution starts] property, [Routine to run immediately after execution stops] property, and [Work RAM start address for executing a specified routine] property, you can also specify function name (when using C language) or label name (when using Assembly language) as property values.

Caution 2. Use an interrupt stack when using a stack inside the specified routine.

Caution 3. Describe an RTS (return from subroutine) instruction for terminating the specified routine processing.

Caution 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.

Caution 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.

Caution 6. Immediately before execution or immediately after stop of the specified routine, execution enters a stopped state under control of the debug tool. The stop period is determined as follows.

- User program execution starts about 100 cycles (about 1 μs when CPU clock is 100 MHz) after the specified routine is executed before execution of the user program.

- Execution of the specified routine starts about 100 cycles (about 1 μs when CPU clock is 100 MHz) after the stop of user program execution.
  In on-chip ROM disabled extended mode, execution of the specified routine starts about 3 ms with a CPU clock of 100 MHz, or about 8 ms with a CPU clock of 32 kHz, after the stop of user program execution. This also applies when the user program stops for the first time after the hot plug-in connection is established.

Caution 7. Execution of a specified routine starts in the supervisor mode. Do not switch it to the user mode.

Caution 8. When executing a specified routine, do not perform memory access/download/break point setting to the program area of the specified routine.

Caution 9. While the facility to execute a specified routine is in use, 230h (560) bytes of RAM starting from [Work RAM start address for executing a specified routine] is reserved for exclusive use by the emulator.
This area must not be used by the user program or routine.
Also be sure to allocate this area to the internal RAM area.
The value of [Work RAM start address for executing a specified routine] must be a multiple of four.

Caution 10. 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	Setting the user stack pointer (USP) 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.

Caution 11. While the specified routine is in execution, the trace, break, real-time RAM monitor and timer measurement will be disabled.

Caution 12. Non-maskable interrupt is always prohibited while the specified routine is in execution.

Caution 13. 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.

Caution 14. If a stack is to be used in the specified routine, the ISP register must be set up in the routine. When the specified routine is started, the value of the ISP register is that of [Work RAM start address for executing a specified routine].

Caution 15. Once the execution of the specified routine is completed, the value of the ISP register must be returned to the value at the start of the execution.

Caution 16. While the facility to execute a specified routine is in use, do not input a reset signal through the reset pin or allow the watchdog timer to reset the CPU.

Caution 17. While the facility to execute a specified routine is in use, do not disable the internal RAM area.

Caution 18. While the specified routine is running, do not set the U flag to 1 (i.e. setting the user stack pointer, or USP).

Caution 19. While the specified routine is running, do not use RTFI (return from fast interrupt) instructions.

Caution 20. Do not set breakpoints or cause break events, trace start events, or trace end events in the specified routine.
If a break event occurs within the routine to run before execution of the user program, for example, the program stops immediately after it has started running. If a trace start event occurs within the routine, no

trace data is acquired while the routine is running. Trace acquisition is not resumed until the user program starts running. The emulator will not behave as expected.

Caution 21. While the facility to execute a specified routine is in use, if the Start function does not shift to the user program, or the Stop function does not stop, an error message "A timeout error has occurred in Start/Stop function processing. The system was reset." is displayed. The emulator is initialized and the user program stops. After the system is reset, trace records are initialized too. You can continue with debugging.

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.

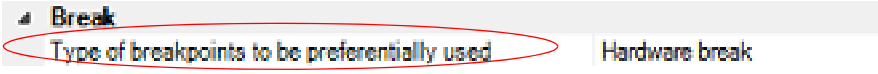
2.10.2.1 Set the type of breakpoints/break timing to use [E1] [E20] [EZ Emulator]
2.10.2.2 Set a breakpoint
2.10.2.3 Edit a hardware breakpoint
2.10.2.4 Delete a breakpoint

Caution [E1/E20/EZ Emulator [RX630, RX631, RX63N, RX63T, RX210, RX110, RX111, RX113, RX634, RX130, and RX13T Groups]]
If a break is generated by an execution-related break that is set to the WAIT instruction or an instruction immediately preceding it, an error may occur. To cause the target program to break at the WAIT instruction or an instruction immediately preceding it, use a breakpoint.

2.10.2.1 Set the type of breakpoints/break timing to use [E1] [E20] [EZ Emulator]

The type of breakpoints/break timing to use can be set in the [Break] [E1] [E20] [EZ Emulator] category on the [Debug Tool Settings] tab of the Property panel.

Figure 2.106 [Break] Category [E1] [E20] [EZ Emulator]



Specify the type of preferential breakpoint (breakpoint that can be set with a single click of the mouse) in the [Type of breakpoints to be preferentially used] property.
Note, however, that if the number of the set breakpoints of the specified type exceeds the limit settable (see "2.17.7.1 Allowable 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 handled as a hardware break event.

Note 1. Hardware break is a "before execution" (, or pre-execution) break as the program will break before executing instruction at the specified address. This function is implemented using the debug tool resources.

Note 2. Software break events cannot be set in external address spaces.

2.10.2.2 Set a breakpoint

Perform this operation in the Editor panel/Disassemble panel in which the source text/disassembled text is displayed.

In the main area of the Editor panel or the event area of the Disassemble panel, 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.

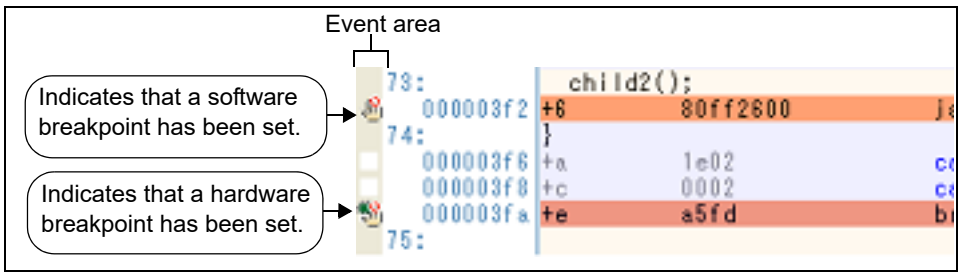
When a breakpoint is set, the following event mark appears at the breakpoint location, and the source text line/ disassembled text line is highlighted.

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 2.4 Event Marks of the Breakpoint

Event Type	Event Mark
Software Break (**[E1] [E20] [EZ Emulator]**)	
Hardware Break	

Figure 2.107 Breakpoint Setting Example (Disassemble Panel)



Caution 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 set. Also see "2.17.1 Changing states of setting (Enabled/Disabled)" for details on breakpoint settings, including the allowable number of valid events.

Caution 2. Breakpoints cannot be set to lines with no address indication.

Caution 3. [E1] [E20] [EZ Emulator]
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.

Remark 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.

Remark 2. **[Simulator]**
Only hardware breakpoints can be set.

Remark 3. **[E1] [E20] [EZ Emulator]**
Hardware/software breakpoints can be set regardless of the specification made in "2.10.2.1 Set the type of breakpoints/break timing to use [E1] [E20] [EZ Emulator]" by the operation described 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.

2.10.2.3 Edit a hardware breakpoint

When editing a hardware break point you have set, first open the Events panel by selecting [View] menu >> [Event]. After selecting the hardware break point you wish to edit on the Events panel, click [Edit Condition...] on 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 "2.17.4.1 Editing execution-related events".

2.10.2.4 Delete a breakpoint

Click event marks displayed in the Editor panel/Disassemble panel to delete breakpoints you have set (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 breakpoint you wish to delete on the Events 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] [EZ Emulator]

The program in execution can be stopped at the arbitrary position by setting a break event (execution-related). It is an "after execution" (, or post-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.

2.10.3.1 Set a beak event (execution-related)
2.10.3.2 Edit a break event (execution-related)
2.10.3.3 Delete a break event (execution-related)

2.10.3.1 Set a beak event (execution-related)

Perform this operation in the Editor panel/Disassemble panel in which the source text/disassembled text is displayed.

After moving the caret to the target line, select [Break Settings] >> [Set Combination Break].

A break event (execution-related) is set to the instruction at the start address corresponding to the clicked line. When a break event (execution-related) is set, the event mark identical to that of hardware breakpoint appears, and the disassembled text line will be highlighted (see "2.10.2.2 Set a breakpoint").

When you have performed this operation, it is interpreted as if a break event (execution-related) 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-related), also see "2.17.7 Points to note regarding event setting" for details on execution-related break event settings, including the allowable number of valid events.

Figure 2.108 Example of Setting a Break Event (Execution-related) on a Line in the Disassembled Text [E1] [E20] [EZ Emulator]

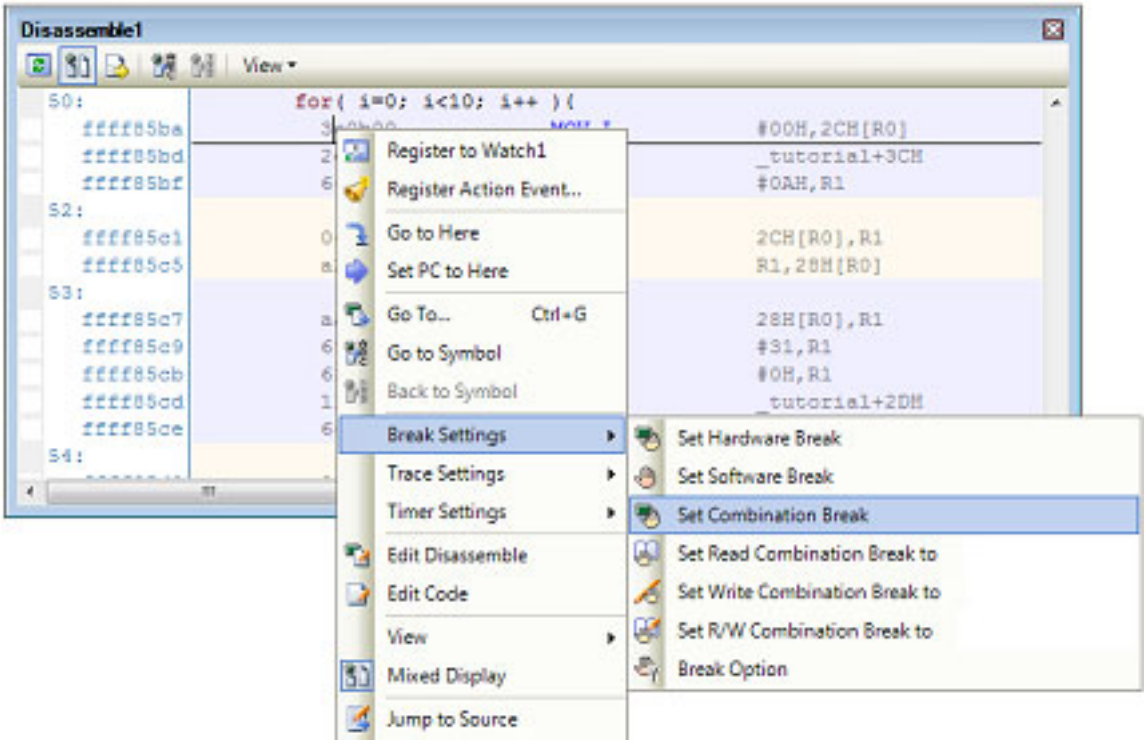
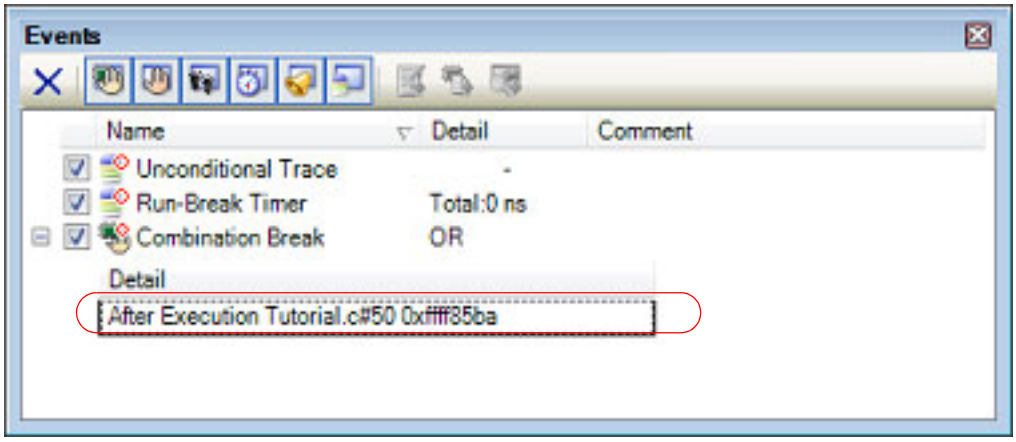


Figure 2.109 Example of a Break Event (Execution-related) in the Events Panel [E1] [E20] [EZ Emulator]



2.10.3.2 Edit a break event (execution-related)

When editing a break event (execution-related) you have set, first open the Events panel by selecting [View] menu >> [Event]. After selecting the break event (execution-related) displayed in the detailed information on the combination break in the Events panel, click [Edit Condition...] on 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 "2.17.4.1 Editing execution-related events".

2.10.3.3 Delete a break event (execution-related)

When deleting a break event you have set, first open the Events panel by selecting [View] menu >> [Event]. After selecting the break event (execution-related) displayed in the detailed information on the combination break in the Events panel, click ( ) button on the toolbar to delete it (see "2.17.5 Deleting events").

Each break event (execution-related) can also be deleted by clicking on the event mark on the source or disassembled text.

2.10.4 Stop the program with the access to variables/I/O registers

By setting a break event (access-related), it is possible to stop the program in execution 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 2.5 Types of Accesses to Variables

Access Type	Description
Read	The program in execution is stopped when a specified variable or I/O register is accessed for read (i.e., to read data from it).
Write	The program in execution is stopped when a specified variable or I/O register is accessed for write (i.e., to write data to it).
Read/Write	The program in execution is stopped when a specified variable or I/O register is accessed for read or write (i.e., to read or write data).

Caution 1. The program is not stopped with the access via DMAC (Dynamic Memory Access Controller) / DTC (Data Transfer Controller).

Caution 2. [Simulator]
With string manipulation instructions and multiply-and-accumulate instructions, the first and final data accesses are subjected to the event check.

This section describes the following.

2.10.4.1 Set a break event (access-related) to a variable/I/O register

2.10.4.2 Edit a break event (access-related) to a variable/IO register
2.10.4.3 Delete a break event (access-related) to a variable/IO register

2.10.4.1 Set a break event (access-related) to a variable/I/O register

Use one of the following methods to set a break event (access-related) that stops programs with the access to a variable/I/O register.

Caution 1. Also see "2.17.7 Points to note regarding event setting" for details on break event (access-related) settings, including the allowable number of valid events.

Caution 2. [E1] [E20] [EZ Emulator]
Break events (access-related) are displayed as Read, Write, or Read/Write in the detailed information on the combination break in the Events panel.

(1) Set a break event (access-related) 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/disassembled 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 static variables inside files can be used.
By performing the following operation, it is interpreted as if a break event (access-related) 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	[E1] [E20] [EZ Emulator] Select [Break Settings] >> [Set Read Combination Break to], and then press the [Enter] key. [Simulator] Select [Break Settings] >> [Set Read Break to], and then press the [Enter] key. If you have specified a value in the text box in the menu, a 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.
Write	[E1] [E20] [EZ Emulator] Select [Break Settings] >> [Set Write Combination Break to], and then press the [Enter] key. [Simulator] Select [Break Settings] >> [Set Write Break to], and then press the [Enter] key. If you have specified a value in the text box in the menu, a 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.
Read/Write	[E1] [E20] [EZ Emulator] Select [Break Settings] >> [Set R/W Combination Break to], and then press the [Enter] key. [Simulator] Select [Break Settings] >> [Set R/W Break to], and then press the [Enter] key. If you have specified a value in the text box in the menu, a break will occur only when the specified value is used for the reading or writing. On the other hand, if no value is specified, reading or writing the selected variable by any value will cause the break to occur.

Caution 1. Variables within the current scope can be specified.

Caution 2. Variables or I/O register at lines that have no valid addresses cannot be used for break events.

Figure 2.110 Example of Setting a Break Event (Access-related) on Variable in the Source Text [E1] [E20] [EZ Emulator]

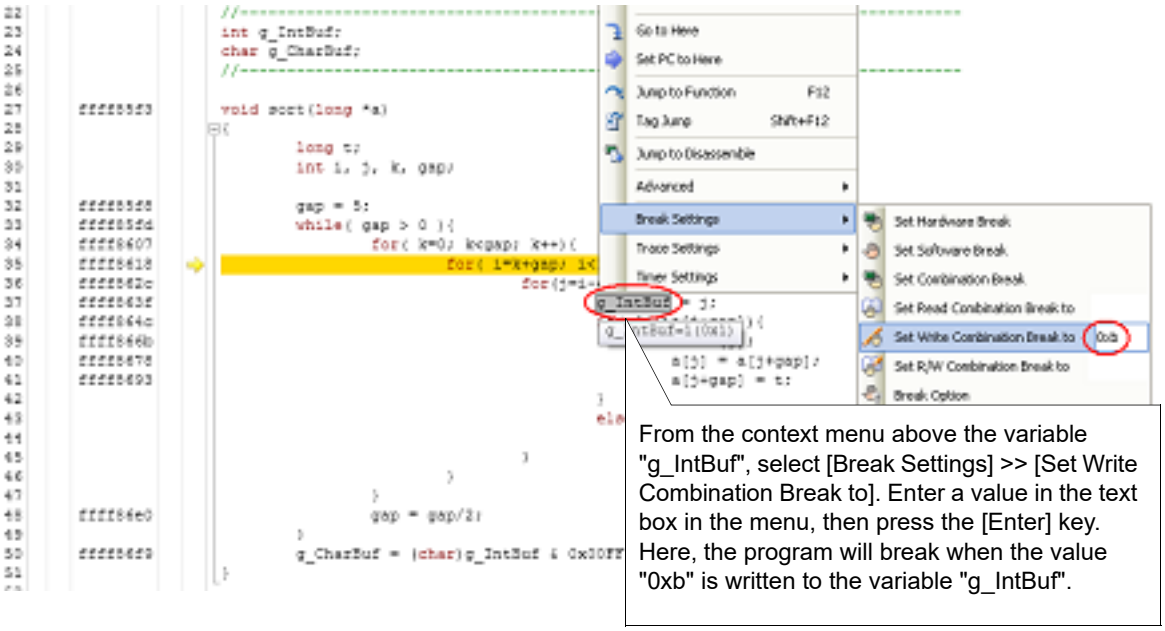


Figure 2.111 Example of a Break Event (Access-related) in the Events Panel [E1] [E20] [EZ Emulator]

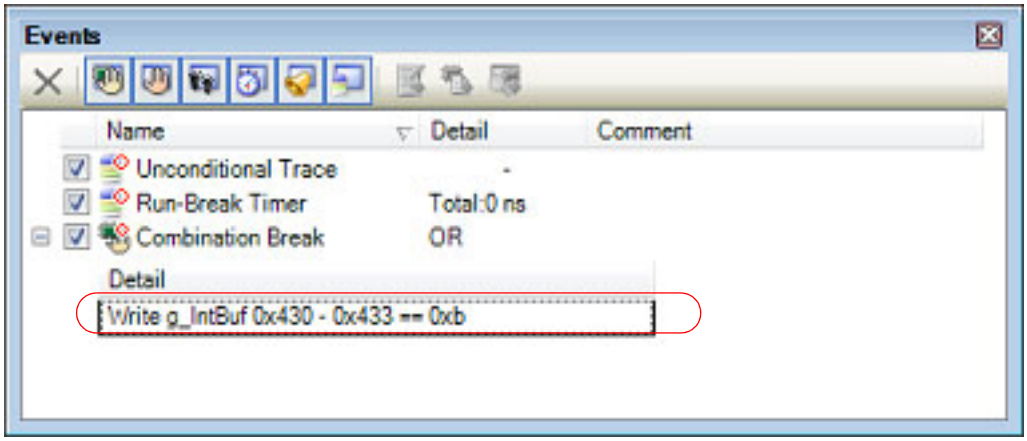


Figure 2.112 Example of Setting a Break Event (Access-related) on a Variable in the Source Text [Simulator]

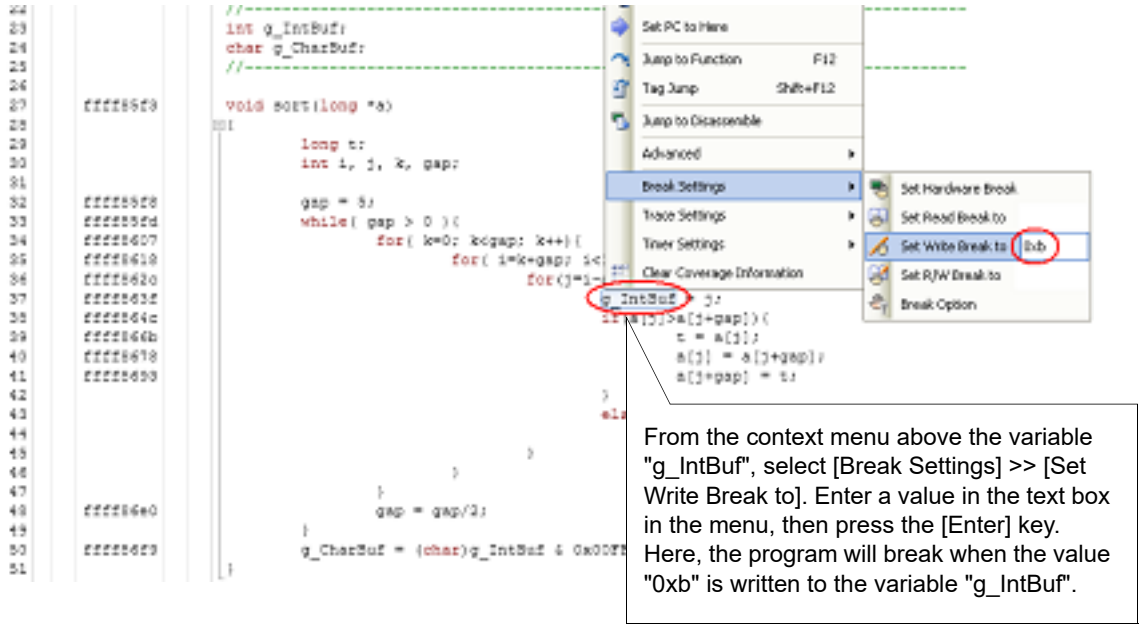
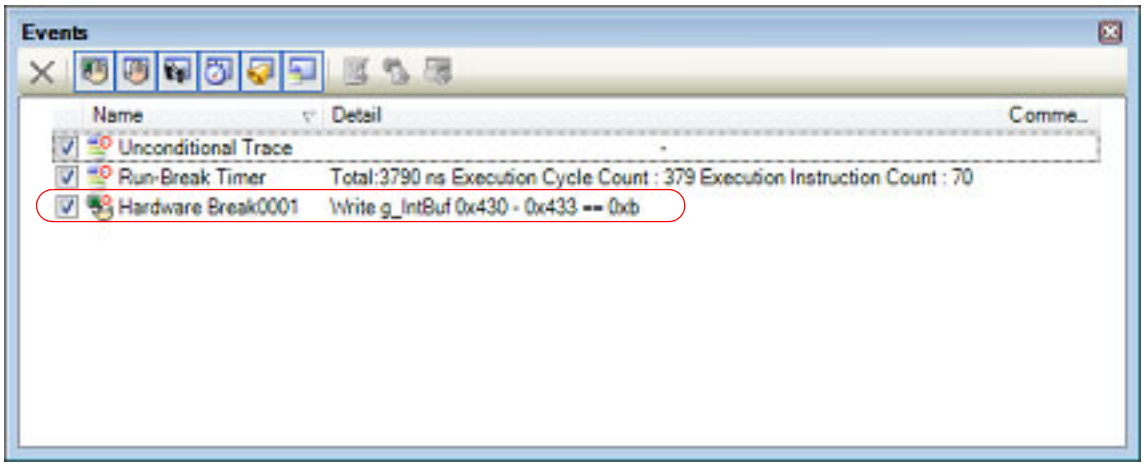


Figure 2.113 Example of a Break Event (Access-related) in the Events Panel [Simulator]



(2) Set a break event (access-related) 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, static variables inside files, and I/O register can be used.

By performing the following operation, it is interpreted as if a break event (access-related) 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	[E1] [E20] [EZ Emulator] Select [Access Break] >> [Set Read Combination Break to], and then press the [Enter] key. [Simulator] Select [Access Break] >> [Set Read Break to], and then press the [Enter] key. If you have specified a value in the text box in the menu, a 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.

Access Type	Operation
Write	[E1] [E20] [EZ Emulator] Select [Access Break] >> [Set Write Combination Break to], and then press the [Enter] key. [Simulator] Select [Access Break] >> [Set Write Break to], and then press the [Enter] key. If you have specified a value in the text box in the menu, a 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.
Read/Write	[E1] [E20] [EZ Emulator] Select [Access Break] >> [Set R/W Combination Break to], and then press the [Enter] key. [Simulator] Select [Access Break] >> [Set R/W Break to], and then press the [Enter] key. If you have specified a value in the text box in the menu, a break will occur only when the specified value is used for the reading or writing. On the other hand, if no value is specified, reading or writing the selected watch-expression by any value will cause the break to occur.

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 2.114 Example of Setting a Break Event (Access-related) on a Watch-expression [E1] [E20] [EZ Emulator]

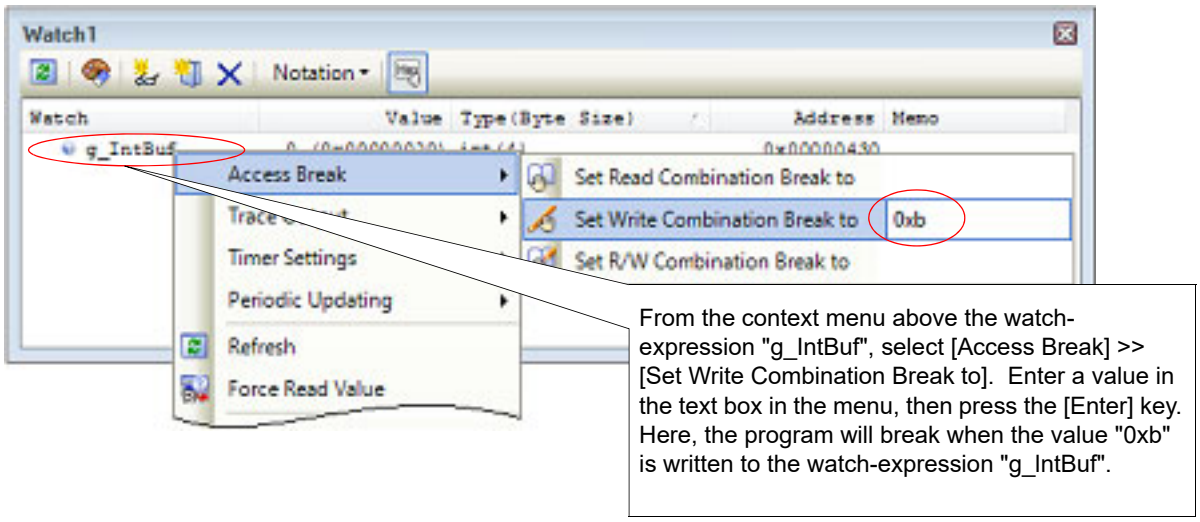
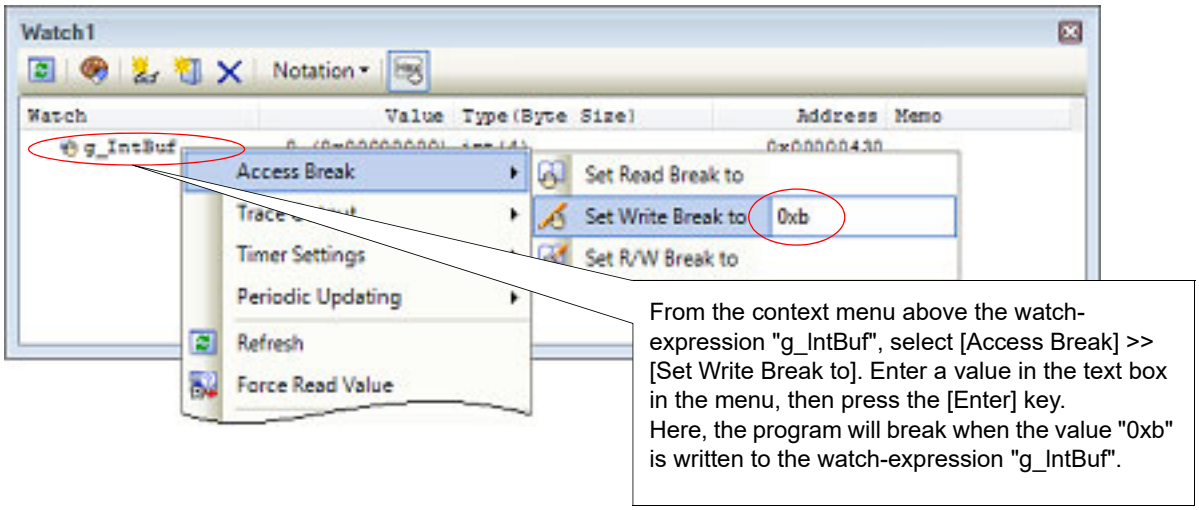


Figure 2.115 Example of Setting a Break Event (Access-related) on a Watch-expression [Simulator]



2.10.4.2 Edit a break event (access-related) to a variable/IO register

When editing a break event (access-related) you have set, first open the Events panel by selecting [View] menu >> [Event]. After selecting the break event (access-related) you wish to edit on the Events panel, click [Edit Condition...] on 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 "2.17.4.2 Editing access-related events".

2.10.4.3 Delete a break event (access-related) to a variable/IO register

When deleting a break event (access-related) 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.

(1) [E1] [E20] [EZ Emulator]
After selecting the break event (access-related) you wish to delete in the detailed information on the combination break, click ( ) button on the toolbar in the above panel (see "2.17.5 Deleting events").
Each break event (access-related) can also be deleted by clicking on the event mark on the disassembled text.

(2) [Simulator]
After selecting the hardware break (access-related) 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] [EZ Emulator]

By setting more than one break event (execution-related and/or access-related) 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 2.6 Combination conditions for the combination break

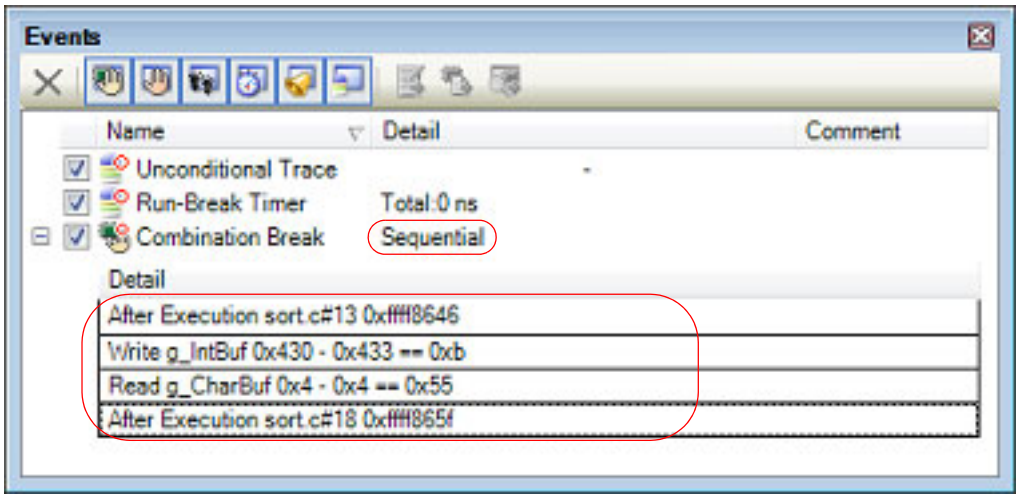
Combination conditions	Description
OR	Break will occur when any one of the break event conditions is satisfied.
AND	Break will occur when all the break event conditions are satisfied irrespective of the time line.
Sequential	Break will occur when the break event conditions are satisfied in the specified 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 "2.17.4.3 Editing combination conditions of events [E1] [E20] [EZ Emulator]".

Caution 1. Also see "2.17.7 Points to note regarding event setting" for details on combination break settings, including the allowable number of valid events.

Caution 2. Also see Cautions of "(1) Editing the combination condition" for details on combination conditions.

Figure 2.116 Example of the Combination Break (Sequential) in the Events Panel [E1] [E20] [EZ Emulator]



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 2.7 Other break factors

Break factors	Debug Tool to Use		
	E1(Serial)/E1(JTAG)/ E20(Serial)/E20(JTAG) EZ Emulator	E2	Simulator
Trace memory full[Note 1]	✓	✓	✓
Temporary break occurred	✓	✓	✓
Execution failed or cause unknown	✓	✓	-
WAIT Instruction executed	-	-	✓
Undefined instruction exception encountered	-	-	✓[Note 2]
Privileged instruction exception encountered	-	-	✓[Note 2]
Access exception encountered	-	-	✓[Note 2]
Floating-point exception encountered	-	-	✓[Note 2, 3]
Interrupt encountered	-	-	✓[Note 2]
INT instruction exception encountered	-	-	✓[Note 2]
BRK instruction exception encountered	-	-	✓[Note 2]
Peripheral function simulation error occurred	-	-	✓
Illegal memory access made	-	-	✓[Note 2]
Stream I/O error occurred	-	-	✓
Allocation of coverage memory failed	-	-	✓
Allocation of trace memory failed	-	-	✓
E2 expansion function	-	✓	-
Fully used the storage memory	-	✓	-

Note 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.

Note 2. **[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.

Note 3. **[Simulator]**
A single-precision floating-point exception can be the object of this factor.

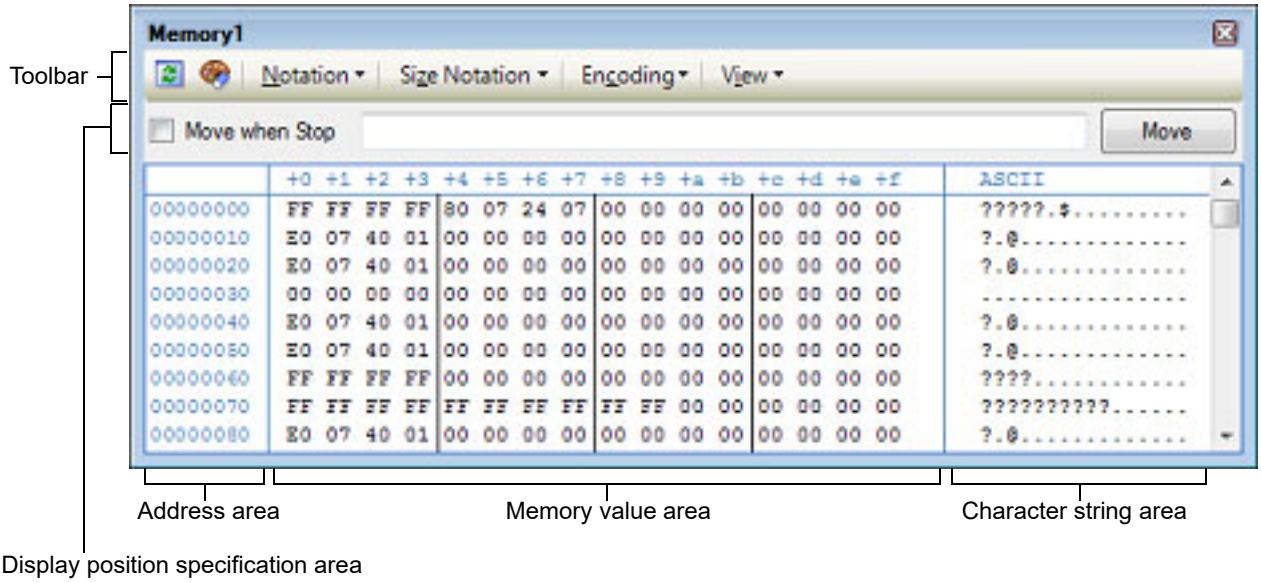
2.11 Displaying and Changing Memory, Registers, and Variables

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 2.117 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.

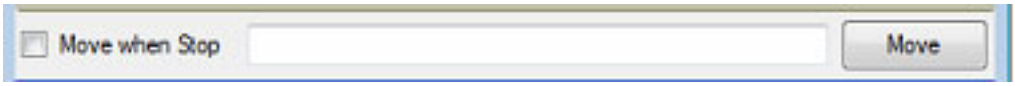
2.11.1.1 Specifying the display position
2.11.1.2 Changing the display form of values
2.11.1.3 Changing memory contents
2.11.1.4 Displaying and changing memory contents during program execution
2.11.1.5 Searching for memory contents
2.11.1.6 Collectively changing (initializing) memory contents
2.11.1.7 Saving displayed memory contents

2.11.1.1 Specifying the display position

By specifying an address expression in the display position specification area, it is possible to specify the position at which CS+ starts displaying memory values. (CS+ starts displaying, by default, from the address "0x0.")

Remark An offset value of the display start position of memory values can be set via the Address Offset Settings dialog box that is opened by selecting [Address Offset Value Settings...] from the context menu.

Figure 2.118 Display Position Specification Area (Memory Panel)



(1) 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.

Remark 1. By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.21.2 Symbol name completion function")

Remark 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.

(2) 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 Stop] check box and by the [Move] button.

[Move when Stop]	☑	The address expression is automatically evaluated after the program has halted and the caret moves to the address derived from that calculation.
	☐	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 Stop] check box is not checked, the address expression is evaluated by clicking this button and the caret moves to the address derived from that calculation.

2.11.1.2 Changing the display form of values

The display format of the address area/memory value area/character strings area can be changed using buttons below on the toolbar.

Notation	Shows the following buttons that change the form in which memory values are displayed.
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 buttons that change the form in which size of memory values are displayed.
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. Values are converted depending on the endian of the target memory area.
4 Bytes	Displays memory values in 32-bit width. Values are converted depending on the endian of the target memory area.
8 Bytes	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 encoding in which character strings are displayed.

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.
Half-Precision Float	Displays character strings as a half-precision floating-point value.
Float	Displays character strings as a single-precision floating-point value[Note].
Double	Displays character strings as a double-precision floating-point value.
Float Complex	Displays character strings as a complex number of single-precision floating-point.
Double Complex	Displays character strings as a complex number of double-precision floating-point.
Float Imaginary	Displays character strings as an imaginary number of single-precision floating-point.
Double Imaginary	Displays character strings as an imaginary number of double-precision floating-point.
View	Shows the following buttons that change the display form.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range.
Column Number Settings...	Opens the Column Number Settings dialog box to set the number of view columns in the memory value area.
Address Offset Value Settings...	Opens the Address Offset Settings dialog box to set an offset value for addresses displayed in the address area.

Note For details on the display of a floating-point value, see the section for the Memory panel.

2.11.1.3 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 code in "ASCII" is specified.

You can edit memory values even when the program is in execution. For details, see "2.11.1.4 Displaying and changing memory contents during program execution".

In the case shown below, caution is required when editing the memory values.

Example 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.

Example 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."

Example 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."

Example 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.

2.11.1.4 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.

Caution 1. [E20(JTAG)[RX600 Series]]
The trace function and the real-time 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 >> [Real-time RAM Monitor]

Caution 2. [E20(JTAG)[RX600 Series]]
If a reset e.g. input of a reset signal through the reset pin, or reset of the CPU by the watchdog timer) is issued while a user program is running, the results of the real-time RAM monitor cannot be guaranteed (values displayed may be incorrect).

Caution 3. [E20]
To use the real-time RAM monitor function, the E20 and the target board must be connected via a 38-pin cable. The real-time RAM monitoring facility is not usable when the E20 and the target board are connected via a 14-pin cable.

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 2.8 through Table 2.10 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, RX700 Series]

Mark	Properties in [Access Memory While Running] Category	Setting Value	Note
B	Access by stopping execution [E1] [E20] [EZ Emulator]	[*No*] (default)	RRM/DMM functions enabled.
	Enable the automatic update of real-time display	[*Yes*] (default)	E20(JTAG) [RX600, RX700 Series]
C	Access by stopping execution [E1] [E20] [EZ Emulator]	[*Yes*]	Pseudo RRM function enabled.
D	Update display during the execution	[*Yes*] (default)	
	Display update interval[ms]	[*Integer number between 100 and 65500*]	

(1) For Read

Table 2.8 Subject Area of the Real-Time Display Update Function (Read by Pseudo-RRM Function)

Area	E1(Serial)/E1(JTAG)/E20(Serial)/E20(JTAG)/EZ Emulator	
	Setting	Subject
Internal ROM	C+D	Entire range
Internal RAM		
Data flash		
Target memory		
Emulation memory	-	
CPU register	Not available	
I/O register	C+D	Entire range

Table 2.9 Subject Area of the Real-Time Display Update Function (Read by RRM Function)

Area	E20(JTAG) [RX600, RX700 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	A+B+D	Automatic setting area Note	D	Entire range

Note **[E20(JTAG) [RX600, RX700 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, CS+ 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.

(2) For write

Table 2.10 Subject Area of the Real-Time Display Update Function (Write by DMM Function)

Area	E1(Serial)/E1(JTAG)/ E20(Serial)/E20(JTAG)/EZ Emulator		E20(JTAG) [RX600, RX700 Series]		Simulator	
	Setting	Subject	Setting	Subject	Setting	Subject
Internal ROM	Not available		Not available		D	Entire range
Internal RAM	C+D	Entire range	A+B+D	Automatic setting area Note		
Data flash	Not available		Not available		-	
Target memory	C+D	Entire range	A+B+D	Automatic setting area Note		
Emulation memory	-		-		D	Entire range
CPU register	Impossible		Impossible		-	
I/O register	C+D	Entire range	A+B+D	Automatic setting area Note	D	Entire range

Note **[E20(JTAG) [RX600, RX700 Series]]**
The target area of the DMM function is the same as the target area of the RRM function.

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 "2.11.1.3 Changing memory contents" and "2.11.6.6 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[Note]	00 00 00 00

Note The watch-expression value will be displayed as "?" in the event of "Lost" on the Watch panel.

Figure 2.119 Example of Memory Display by the Pseudo-RRM Function (Memory Panel) [E1] [E20] [EZ Emulator]

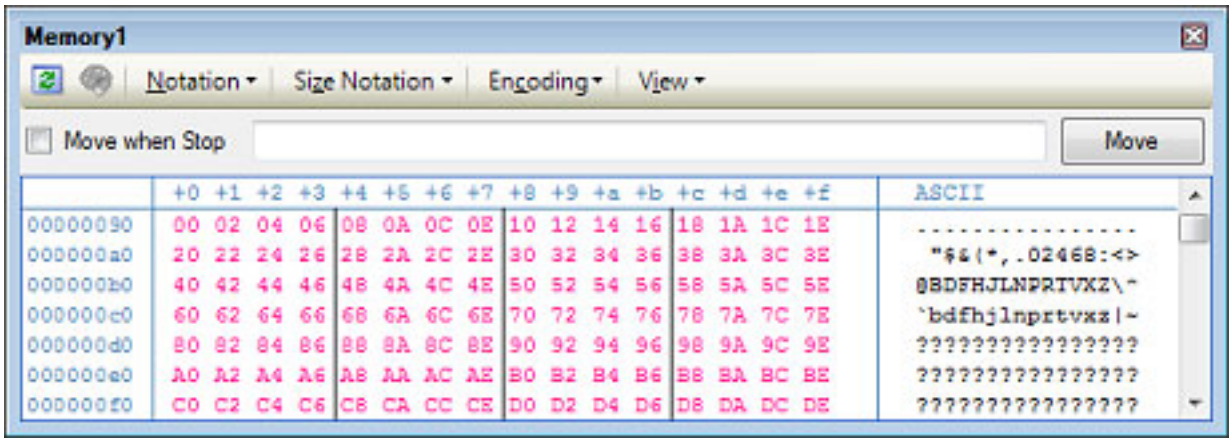


Figure 2.120 Example of Memory Display by the RRM Function (Memory Panel) [Simulator]

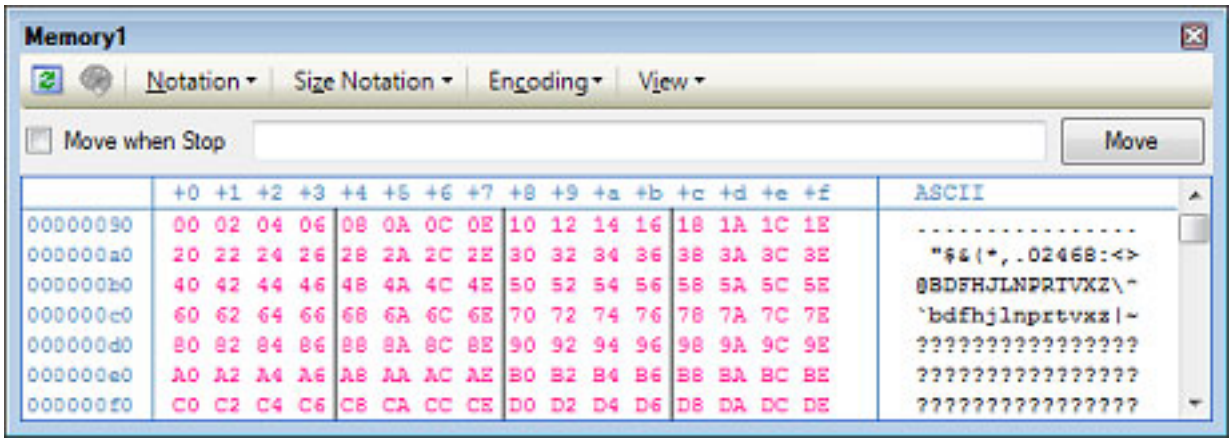
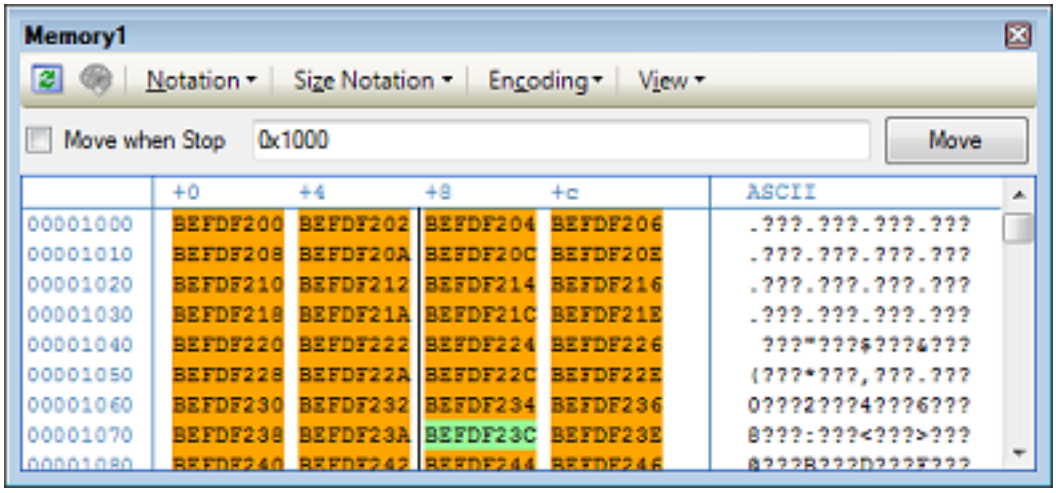


Figure 2.121 Example of Memory Display by the RRM Function (Memory Panel) [E20 (JTAG) [RX600 Series]

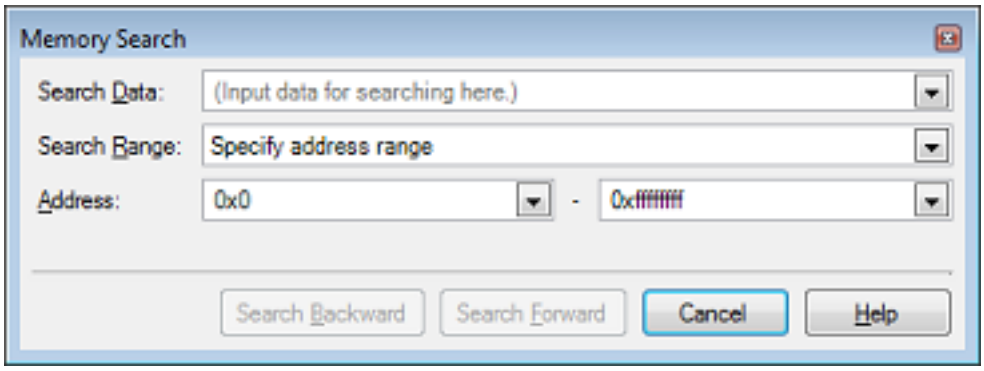


2.11.1.5 Searching for memory contents

To search for memory values, use the Memory Search dialog box that is opened by selecting [Find…] from the 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 2.122 Searching for Memory Contents (Memory Search Dialog Box)



Caution 1. Memory contents cannot be searched during program execution.

Caution 2. Character strings displayed as floating-point values cannot be searched.

(1) 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 encoding 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.

(2) Specifying the [Search Range]
Select the 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].
Memory mapping	A search is conducted within the selected range of memory mapping. This list item displays memory mappings individually (except non-mapped areas) that are displayed in the Memory Mapping dialog box. Display form: *<memory type> <address range> <size>*

(3) Specifying the [Address]
This item is valid only when you've selected [Specify Address Range] in "(2) 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.

Remark 1. By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.21.2 Symbol name completion function").

Remark 2. If the "start address" text box is blank, address "0x0" is assumed.

Remark 3. If the "end address" text box is blank, the upper-limit address of the microcontroller's address space is assumed.

(4) 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.

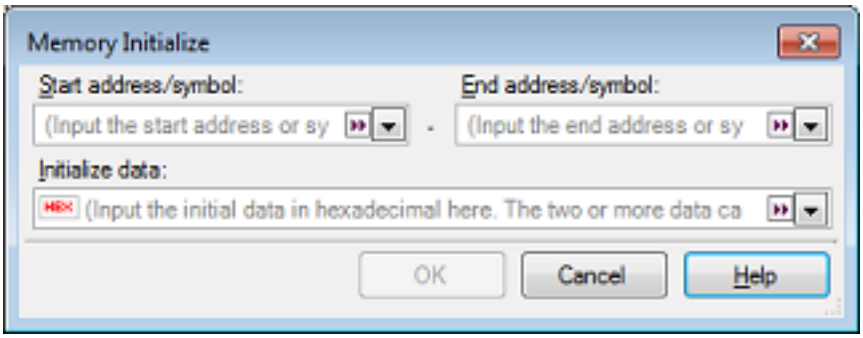
2.11.1.6 Collectively changing (initializing) memory contents

Memory values can be changed collectively (initialized).

Selecting [Fill...] from the 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 2.123 Changing Memory Contents Collectively (Memory Initialize Dialog Box)



(1) 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 "2.21.2 Symbol name completion function").

(2) 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.

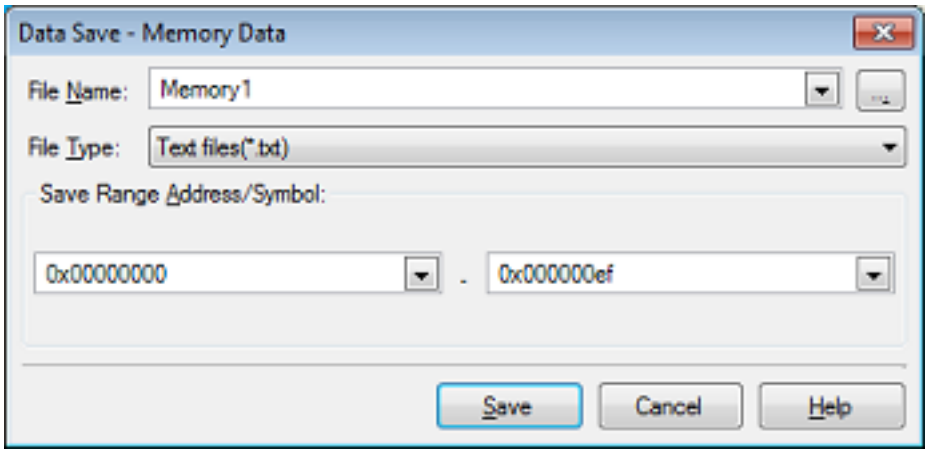
Input Character String (Initialize data)	Written-in image (in bytes)	
	Little endian	Big endian
1	01	01
0 12	00 12	00 12
00 012 345	00 12 00 45 03	00 00 12 03 45
000 12 000345	00 00 12 45 03 00	00 00 12 00 03 45

(3) 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, CS+ displays a message and does not initialize memory values.

2.11.1.7 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, CS+ 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 you perform this operation while a range is selected on panel, 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 2.124 Saving Memory Data (Data Save Dialog Box)



(1) 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.

(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 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.

(3) 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 "2.21.2 Symbol name completion function").

(4) Clicking the [Save] button
Memory data is saved in specified form to a specified file.

Figure 2.125 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)

```
00000000,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,................
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
```

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.

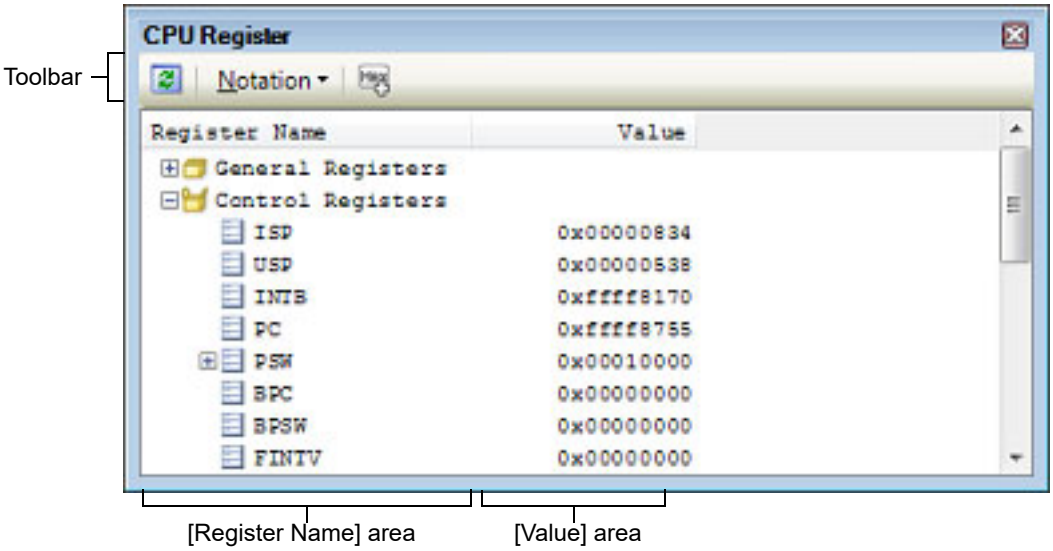
2.11.2 Displaying and changing the CPU registers

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 2.126 Displaying the CPU Register Contents (CPU Register Panel)



Following methods of operation are described here.

2.11.2.1 Changing the form in which values are displayed
2.11.2.2 Changing the CPU register contents
2.11.2.3 Saving the displayed CPU register contents

2.11.2.1 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.

AutoSelect	Displays the value of a selected item (including low-order item) in predetermined notation (default).
Hexadecimal	Displays the value of a selected item (including low-order item) in hexadecimal.
Signed Decimal	Displays the value of a selected item (including low-order item) in signed decimal.
Unsigned Decimal	Displays the value of a selected item (including low-order item) in unsigned decimal.
Octal	Displays the value of a selected item (including low-order item) in octal.
Binary	Displays the value of the selected item (including sub-items) in binary.
ASCII	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.
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 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 ( ).

2.11.2.2 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.

2.11.2.3 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, CS+ gets latest information from the debug tool.

Figure 2.127 CPU Register Value Output Image When Saved

```
Register name           Value
-----------------------------------------------------------------------
Category name
 -Register name         Value
      :                   :
```

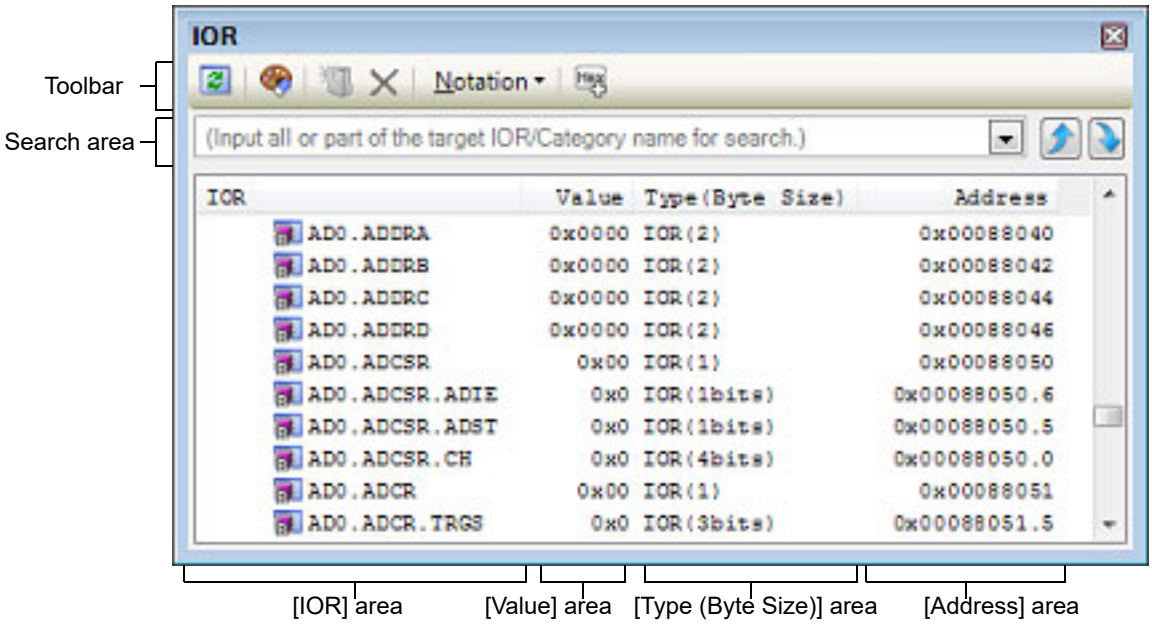
2.11.3 Displaying and changing the I/O registers

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 2.128 Displaying the I/O Register Contents (IOR Panel)



Following methods of operation are described here.

2.11.3.1 Searching I/O register
2.11.3.2 Putting the I/O registers in order
2.11.3.3 Changing the form in which values are displayed
2.11.3.4 Changing the contents of I/O registers
2.11.3.5 Displaying and changing I/O register contents during program execution
2.11.3.6 Saving the displayed I/O register contents

2.11.3.1 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.

Remark 1. I/O register names which are categorized in folders and hidden are also searched (They will be expanded and selected.).

Remark 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.

2.11.3.2 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).

Caution 1. No other categories can be created within a category.

Caution 2. No I/O registers can be added or removed.

(1) 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.

(2) 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.

(3) To remove a category
Select a category you want to remove and then click the toolbar button However, only blank categories can be removed.

(4) 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.

2.11.3.3 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.
Hexadecimal	Displays the value of a selected item in hexadecimal (default).
Signed Decimal	Displays the value of a selected item in signed decimal.
Unsigned Decimal	Displays the value of a selected item in unsigned decimal.
Octal	Displays the value of a selected item in octal.
Binary	Displays the value of a selected item in binary.
ASCII	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 ( ).

2.11.3.4 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.

Caution 1. This operation cannot be performed during program execution.

Caution 2. The values of read-only I/O registers cannot be changed.

Remark 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.

Remark 2. If a number with more digits than the size of the I/O register is entered, the higher-order digits will be masked.

Remark 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".

2.11.3.5 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".

2.11.3.6 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, CS+ 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 2.129 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 Variable] from the [View] menu.

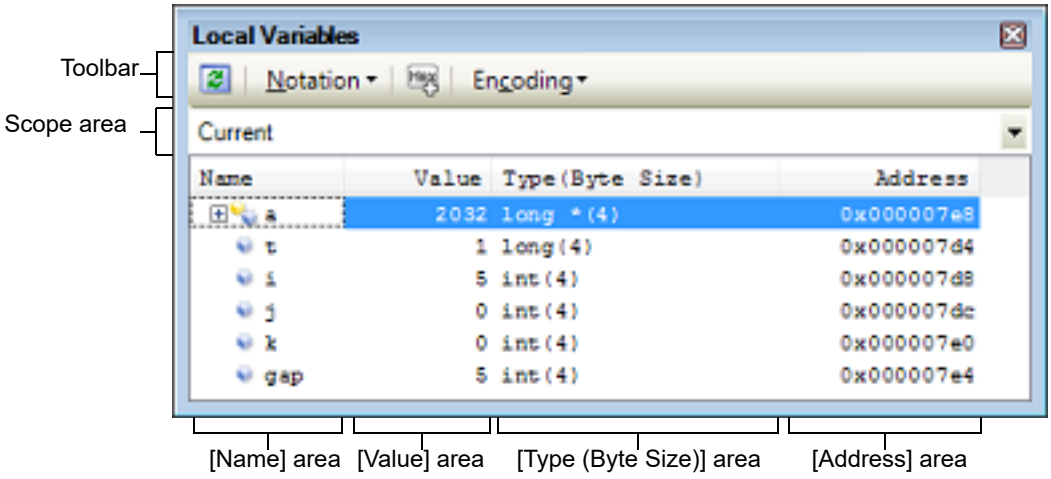
To display the content of your desired local variable, select a scope in the scope area.

The Local Variables 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.
The contents of each area on this panel are displayed at the time the program has stopped running.

Figure 2.130 Displaying the Contents of Local Variables (Local Variables Panel)



Following methods of operation are described here.

2.11.5.1 Changing the form in which values are displayed
2.11.5.2 Changing the contents of local variables

2.11.5.3 Saving the displayed contents of local variables

2.11.5.1 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.
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 in decimal (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel in hexadecimal.
Float	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.
Double	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 encoding in which string variables 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.

2.11.5.2 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, CS+ 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.

Remark 1. If the numeral entered for a variable is smaller than the size of the variable, its high-order digits are padded with zeroes.

Remark 2. If the numeral entered for a variable is larger than the size of the variable, its high-order digits are masked.

Remark 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.

Remark 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.

2.11.5.3 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, CS+ 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 2.131 Local Variable Value Output Image When Saved

```
Scope : Current scope
[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 "2.11.6.7 Displaying and changing the contents of watch-expressions during program execution").

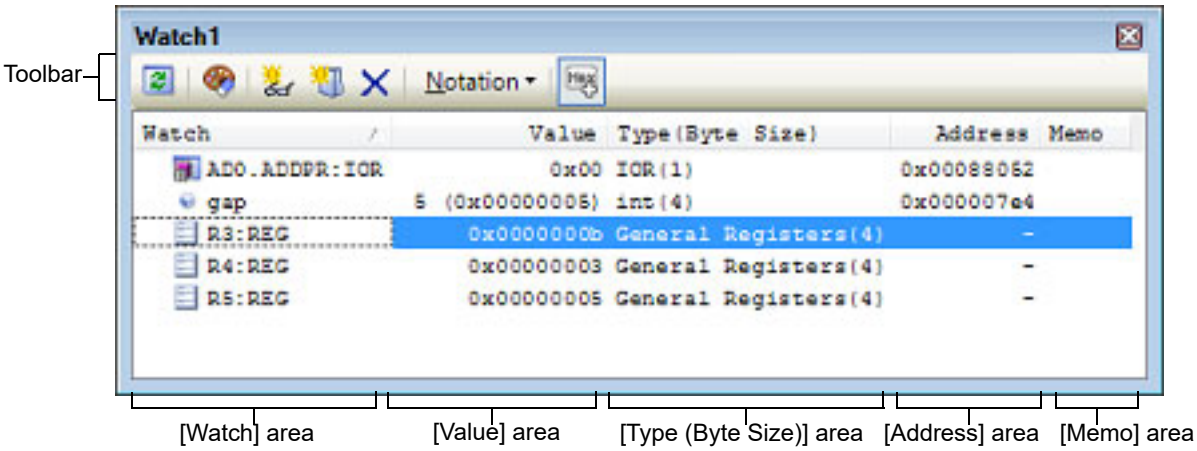
The Watch panel is opened by choosing [Watch] from the [View] menu and then selecting [Watch 1-4].

Up to four pieces of Watch panels can be opened at a time. Each panel is discriminated by the name "Watch1," "Watch2," "Watch3," and "Watch4" 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.

Caution It is not possible to display or change the CPU register contents during program execution.

Figure 2.132 Displaying the Contents of Watch-expression (Watch panel)



Following methods of operation are described here.

2.11.6.1 Registering watch-expressions
2.11.6.2 Putting the registered watch-expressions in order
2.11.6.3 Editing the registered watch-expressions
2.11.6.4 Removing watch-expressions
2.11.6.5 Changing the form in which values are displayed
2.11.6.6 Changing the contents of watch-expressions
2.11.6.7 Displaying and changing the contents of watch-expressions during program execution
2.11.6.8 Exporting/importing watch-expressions
2.11.6.9 Saving the displayed contents of watch-expressions

2.11.6.1 Registering watch-expressions

There are following three methods to register watch-expressions (By default, no watch-expressions are registered.).

Caution 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).

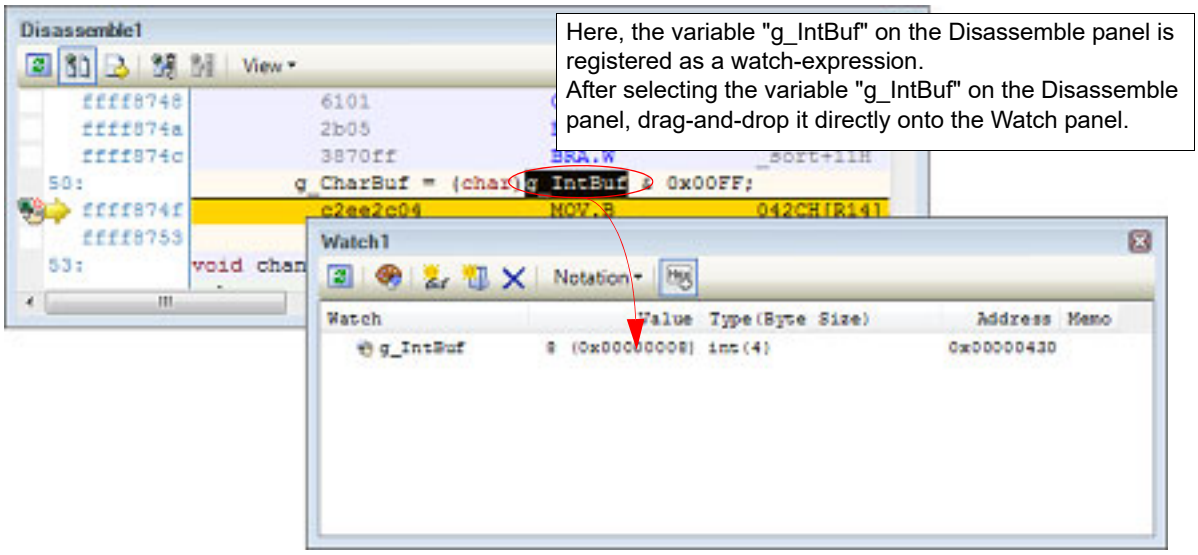
Caution 2. Because of optimization by a compiler, for blocks where variables, or the subject to be operated on, are not in use, there may be no variable data in the stack and registers. In this case, those unused variables, even when registered as watch-expressions, will have their displayed values marked with "?".

Remark 1. The respective watch-expressions registered in each Watch panel (Watch1 through Watch4) are managed individually and saved as user information for the project.

Remark 2. Plural watch-expressions with the same name can be registered.

(1) To register from other panels
Watch-expressions can be registered from other panels of CS+.
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 (Watch1 through Watch4).
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.5 Relationship between Panels and Subjects Registrable as Watch-expressions".

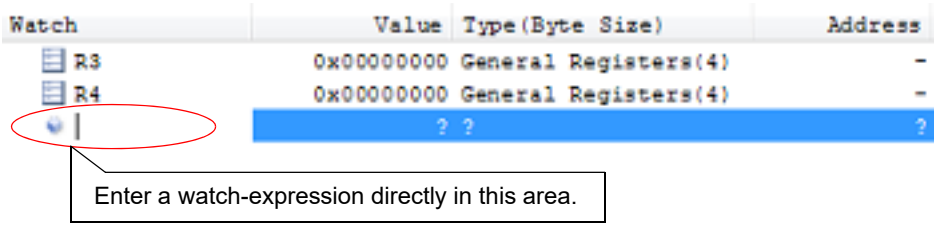
Figure 2.133 Example for Registering Watch-expressions from Other Panels



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 to Watch1] from the context menu (However, this method is usable for only the Watch panel (Watch1)).

(2) To register directly on the Watch panel
In any Watch panel (Watch1 through Watch4), click the toolbar button and the entry box shown below will be displayed in the [Watch] area.

Figure 2.134 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.6 Watch-expression Input Form"
- "Table A.8 Handling of a C Variable when Registered in Watch by Specifying Scope"
- "Table A.9 Handling of a CPU Register when Registered in Watch by Specifying Scope"
- "Table A.10 Handling of an I/O Register when Registered in Watch by Specifying Scope"

Remark By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.21.2 Symbol name completion function").

(3) 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 (Watch1 through Watch4).
In this case, the character string you've dropped is registered as a watch-expression directly as it is.

2.11.6.2 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.).

Caution 1. No other categories can be created within a category.

Caution 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.).

(1) 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.

(2) 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.

(3) To remove a category
Select a category you want to remove and then click the toolbar button .

(4) 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 If categories or watch-expressions in a watch panel are drag-and-dropped into another Watch panel (Watch1 to Watch4), the categories or watch-expressions are copied into the target Watch panel.

2.11.6.3 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.

2.11.6.4 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 .

2.11.6.5 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.
AutoSelect	Displays the value of a selected watch-expression in per-variable predetermined notation (default) (see "Table A.11 Display Form of Watch-expressions (Default)").
Hexadecimal	Displays the value of a selected item in hexadecimal.
Signed Decimal	Displays the value of a selected item in signed decimal.
Unsigned Decimal	Displays the value of a selected item in unsigned decimal.
Octal	Displays the value of a selected item in octal.
Binary	Displays the value of a selected item in binary.
ASCII	Displays the value of a selected item in ASCII code.
Float	Displays a selected item in Float. However, this is valid only when the selected watch-expression consists of 4-byte data.
Double	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.

2.11.6.6 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 "2.11.1.4 Displaying and changing memory contents during program execution" for details on how to operate it.

Remark 1. If the numeral entered for a variable is smaller than the size of the variable, its high-order digits are padded with zeroes.

Remark 2. If the numeral entered for a variable is larger than the size of the variable, its high-order digits are masked.

Remark 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 (FTF-8/UTF-16)) to enter values.

Remark 4. 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.

2.11.6.7 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 "2.11.1.4 Displaying and changing memory contents during program execution"

2.11.6.8 Exporting/importing watch-expressions

This feature allows you to export currently registered watch-expressions to a file and import it so that the watch-expressions can be re-registered.

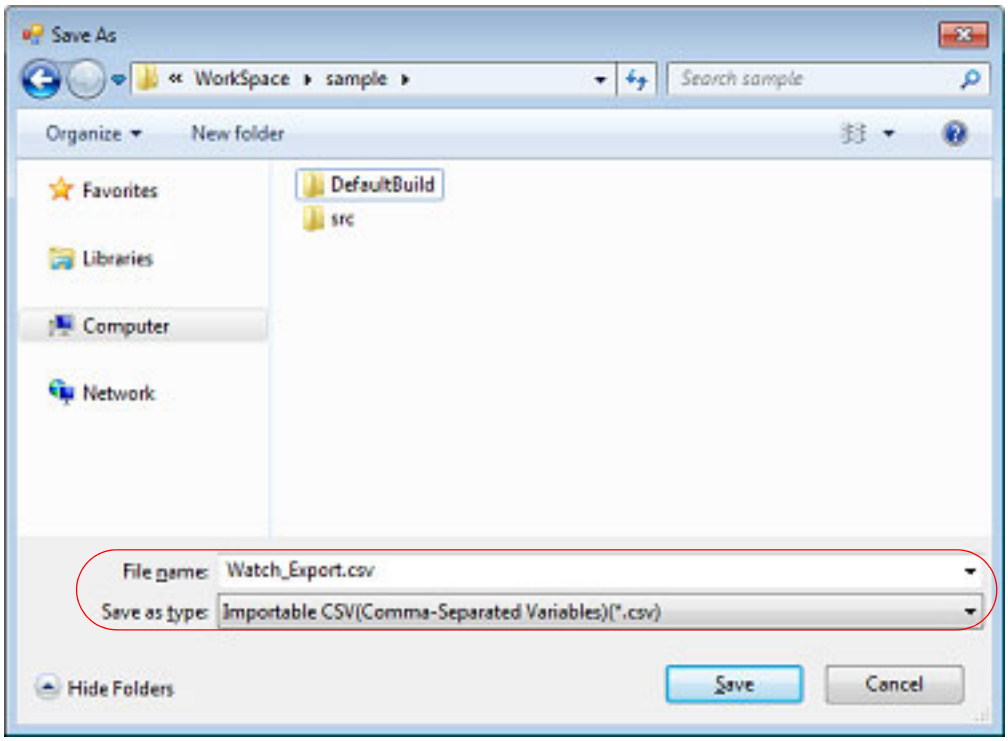
To do this, follow the procedure described below.

(1) Export watch-expressions
Save watch-expressions currently being registered (including categories) in a file format that is possible to import.
While the focus is in the Watch panel, select [Save Watch Data As...] from the [File] menu.
On the Save As dialog box that is automatically opened, specify the following items, and then click the [Save] button.

- [File name]: Specify the name of a file to be saved (the file extension must be "csv").
- [Save as type]: Select "Importable CSV (Comma-Separated Variables)(*.csv)".

Caution Neither values nor the type information of watch-expressions can be saved.
Items that are expanded after analyzing watch-expressions (i.e. an array, structure, and so on) cannot be saved.

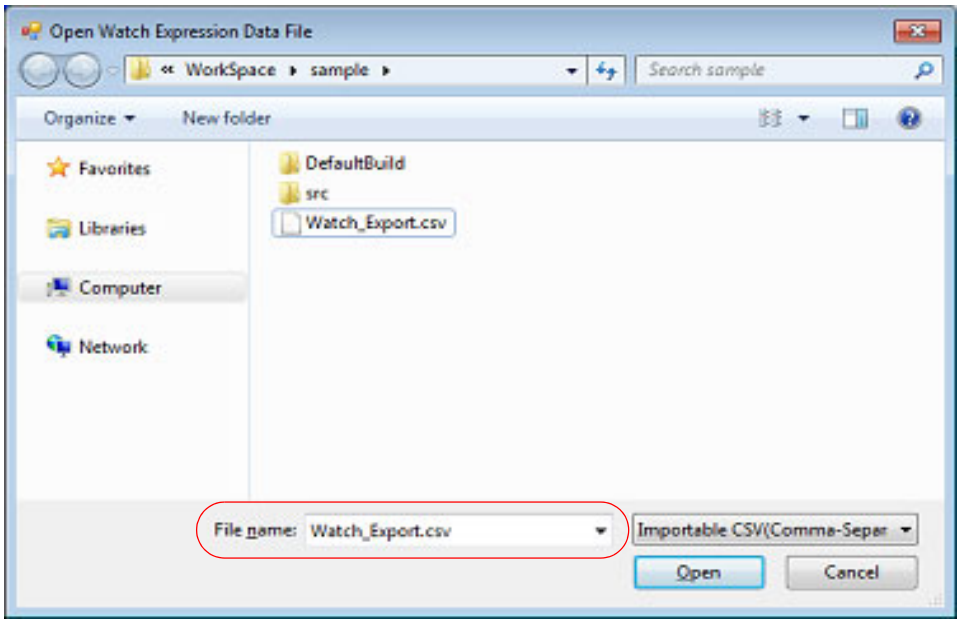
Figure 2.135 Export of Watch-Expressions



(2) Import watch-expressions
Import the file that was exported in (1) to the Watch panel.
On the Watch panel to which you want to import watch-expressions, select [Import Watch Expression...] from the context menu.
On the Open Watch Expression Data File dialog box that is automatically opened, specify the exported file, and then click the [Open] button.

Remark If watch-expressions have been already registered, then imported watch-expressions will be registered at the bottom of them.

Figure 2.136 Import of Watch-Expressions



2.11.6.9 Saving the displayed contents of watch-expressions

By selecting the [File] menu >> [Save Watch Data As...] or selecting [Save Expanded Watch Data…] from the context menu, the Save As dialog box can be opened, and all the contents of the watch-expression and its value can be saved in a text file (*.txt) or CSV file (*.csv).

When saving the contents to the file, all the values of the watch-expression are reacquired and save the latest values acquired.

Note that the values of read-protected I/O register are not re-read. If you want to save the latest values of those, select [Force Read Value] from the context menu then save the file.

Note that for watch-expressions that can be displayed expanded, such as arrays, pointer type variables, structures/unions, and CPU registers (only those with the part name), the behavior differs depending on whether the watch-expression is saved with [Save Watch Data As...] or [Save Expanded Watch Data…].

- When saved with [Save Watch Data As...]
  If arrays, pointer type variables, structures/unions, and CPU registers (only those with the part name) are displayed expanded, the value of each expanded element is also saved. When they are not expanded, "+" mark is added on the top of the item and the value becomes blank.
- When saved with [Save Expanded Watch Data…]
  The watch-expression is expanded up to the maximum 255 nests regardless of the expanded state, and the value of each expanded element is also saved.

Figure 2.137 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 (Watch1-4) are handled individually.

2.12 Display Function Call Information from the Stack

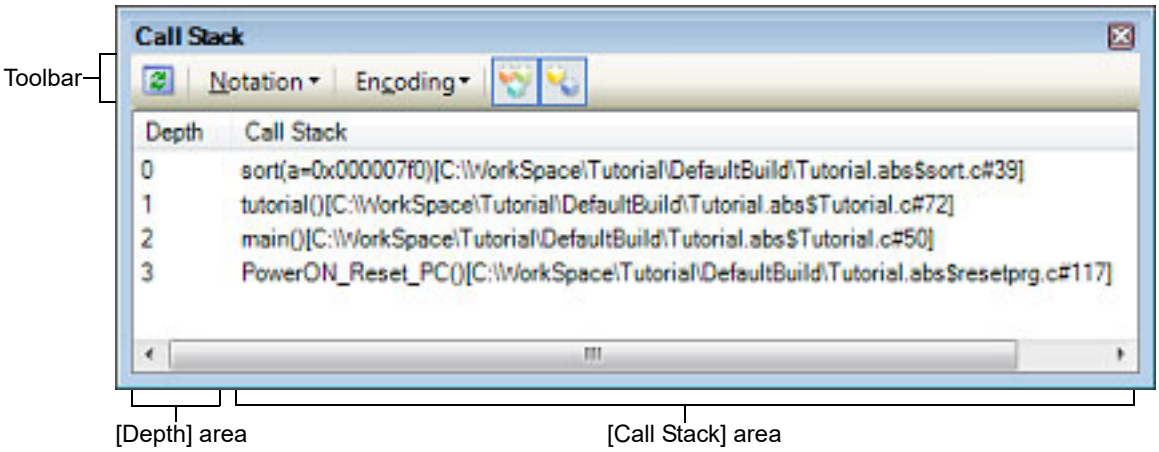
This section describes how to display function-call information from the stack.
The compiler bundled with CS+ (CC-RX) places function-call information on the stack in line with the ANSI standards. By analyzing this function-call information (hereafter referred to as call stack information), it is possible 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.
The contents of each area on this panel are displayed at the time the program has stopped running.

Figure 2.138 Displaying the Call Stack Information (Call Stack Panel)



This section describes the following.

2.12.1.1 Changing the form in which values are displayed
2.12.1.2 Jumping to the source line
2.12.1.3 Displaying local variables
2.12.1.4 Saving the displayed contents of call stack information

2.12.1.1 Changing the form in which values are displayed

The display format of this panel can be changed using buttons below on the toolbar.

Notation	Shows the following buttons that change 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 buttons that change the encoding in which string variables are displayed.
ASCII	Displays string variables on this panel in ASCII code (default).
Shift_JIS	Displays string variables on this panel in Shift_JIS code.
EUC-JP	Displays string variables on this panel in EUC-JP code.
UTF-8	Displays string variables on this panel in UTF-8 code.
UTF-16	Displays string variables on this panel in UTF-16 code.

2.12.1.2 Jumping to the source line

Double-clicking on a line will open the Editor panel, with the caret moved to the source line from which the function indicated by the selected line is called. (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 address from which the function indicated by the selected line is called. (The view jumps to the Disassemble panel (Disassemble1) if it is already open.)

2.12.1.3 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.

2.12.1.4 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 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 2.139 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 explore a process of malfunction until the program starts running wild, providing an effective means for discovering potential bugs in the program.

Caution 1. [E20(JTAG) [RX600, RX700 Series]]
Part of the trace functions and real-time RAM monitor functions (RRM functions) can be used only on a mutually exclusive basis.

Caution 2. [E1] [E20] [EZ Emulator]
When you use a Printf event (an action event) and it occurs, the execution history up to that point is cleared and the onward execution history is recorded.

Caution 3. [EZ Emulator [RX100, RX200 Series]]
If the reset signal is input through the reset pin, all trace data that has been acquired before the reset is erased. Trace acquisition is resumed when the EZ Emulator restarts the execution of the program.

Caution 4. [E1] [E20] [EZ Emulator]
If a reset (e.g. input of a reset signal through the reset pin, or reset of the CPU by the watchdog timer) is issued while a user program is running, the correct recording of trace information both before and after the reset will not be possible.

Caution 5. When trace start events and end events have been set, stopping and restarting tracing during program execution is not possible.

Caution 6. [OCD (JTAG)] [OCD (serial)]
The timestamps of trace information will not indicate the right times if the time between frames exceeds that corresponding to the trace counter (20 bits) or when trace output is lost.

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.

2.13.1.1 For [E1]
2.13.1.2 For [E20]
2.13.1.3 For [EZ Emulator]
2.13.1.4 For [Simulator]

2.13.1.1 For [E1]

Make settings in the [Trace] category on the Property panel's [Debug Tool Settings] tab.

Figure 2.140 [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 + Data access**
Bus master of data access	CPU
Output timestamp	**Yes**
Trace clock count source[MHz]	
Division ratio of trace clock count source	1/1

(1) [Usage of trace function]
Only the trace function can be used.

(2) [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, CS+ continues writing trace data over the old data (default).
Stop trace	When the trace memory is filled, CS+ stops writing trace data (but does not stop program execution).
Stop	When the trace memory is filled, CS+ stops writing trace data and stops program execution at the same time.

(3) [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.

Note 1. **[E1(Serial) [RX100, RX200 Series]]**
Trace data for [Branch + Data access] cannot be collected. Therefore, this item is not displayed in the drop-down list.

Note 2. **[E1(Serial) [RX100, RX200 Series]]**
To collect trace data for [Data access], it is necessary to set address conditions in a point trace. Trace information on data access is not acquired when no address conditions are set. Only records of access to specified addresses are included in the trace information on data access. The start address specified for point-to-point tracing is displayed in the Trace panel.
For details about the point trace, see "2.13.4 Collecting an execution history only when conditions are met".

(4) [Bus master of data access] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
This property is displayed only when [Branch + Data access] or [Data access] is specified in the [Trace data type] property.
Select the Bus master of data access from the drop-down list.
The following bus masters are displayed in the drop-down list.

CPU	The results of data access from CPU are displayed on the trace panel.
DMAC/DTC	The results of data access from DMAC/DTC are displayed on the trace panel.

For data access tracing, only the trace results of data access from the specified bus master are displayed on the trace panel.

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for selecting the Bus master of data access, the [Bus master of data access] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups] property is not displayed. In this case, the bus master is fixed to [CPU].

(5) [Output timestamp]
This property specifies whether timestamp information is added to the trace data to be collected. Specify [Yes] when you want timestamp to be output.

Caution 1. **[E1(Serial) [RX200 Series]]**
If [Branch] is specified in the [Trace data type] property, timestamp cannot be output. If you want it to be output, specify [Data access].

Caution 2. **[E1 [RX100 Series]]**
This property is fixed to [No] because these microcontrollers do not support the output of timestamp information.

(6) [Trace clock count source[MHz]]
Enter a count source for the timestamp in the range 0.0001 to 999.999.
Note that if this property is blank, the set value of the [Operating frequency [MHz]] in the [Clock] category of a [Connect Settings] tab is used in place of the count source.

(7) [Division ratio of trace clock count source] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
This property is displayed only when [Yes] is selected in the [Output timestamp] property.
Select the frequency division ratio for the timestamp count source from the drop-down list.
The following frequency division ratios are displayed in the drop-down list.

1/1	The frequency of the timestamp count source is used without change.
1/16	The frequency of the timestamp count source is multiplied by 1/16 before use.
1/256	The frequency of the timestamp count source is multiplied by 1/256 before use.
1/4096	The frequency of the timestamp count source is multiplied by 1/4096 before use.

The frequency specified in the [Trace clock count source[MHz]] property is divided by the specified value (the frequency is multiplied by 1/n) and one cycle of the obtained frequency is used as the unit for timestamp count (the frequency for count value 1).

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for dividing the timestamp frequency for tracing, the [Division ratio of trace clock count source] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups] property is not displayed. In this case, the division ratio is fixed to [1/1].

2.13.1.2 For [E20]

Make settings in the [Trace] category on the Property panel's [Debug Tool Settings] tab.

Figure 2.141 [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 + Data access (without Access data)**
Start address for trace area without access data	HEX 0
End address for trace area without access data	HEX 3FFFF
Bus master of data access	CPU
External trace output	CPU execution
Trace memory size[MByte]	1
Output timestamp	**Yes**
Trace clock count source[MHz]	
Division ratio of trace clock count source	1/1

(1) [Usage of trace function]
Part of the trace functions and real-time RAM monitor functions (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). - The 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.

Note 1. **[E20(Serial)]**
The real-time RAM monitor function is not supported. Therefore, do not specify [Real-time RAM monitor] for this property value.

Note 2. **[E20(JTAG) [RX600, RX700 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.

(2) [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, CS+ continues writing trace data over the old data (default).
Stop trace	When the trace memory is filled, CS+ stops writing trace data (but does not stop program execution).
Stop	When the trace memory is filled, CS+ stops writing trace data and stops program execution at the same time.

(3) [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.
Branch + Data access (without Access data)[Note 3]	Source/destination address information of branching during program execution, as well as data information on access events that occurred are collected as trace data. However, access data is not collected.
Data access (without Access data)[Note 3]	Data information on access events that occurred during program execution are collected as trace data. However, access data is not collected.

Note 1. **[E20(Serial) [RX100, RX200 Series]]**
Trace data for [Branch + Data access] cannot be collected. Therefore, this item is not displayed in the drop-down list.

Note 2. **[E20(Serial) [RX100, 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".

Note 3. This item can be used only for RX66x, RX65N-2M, RX651-2M, RX671, RX72T, RX72M, and RX72N groups.

(4) [Start address for trace area without access data][RX65N-2M, RX651-2M, RX66T, RX671, RX72T, RX72M, RX66N, and RX72N Group]
Specify the start address for the trace area without access data.

(5) [End address for trace area without access data][RX65N-2M, RX651-2M, RX66T, RX671, RX72T, RX72M, RX66N, and RX72N Group
Specify the end address for the trace area without access data.

(6) [Bus master of data access] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
This property is displayed only when [Branch + Data access] or [Data access] is specified in the [Trace data type] property.
Select the Bus master of data access from the drop-down list.
The following bus masters are displayed in the drop-down list.

CPU	The results of data access from CPU are displayed on the trace panel.
DMAC/DTC	The results of data access from DMAC/DTC are displayed on the trace panel.

For data access tracing, only the trace results of data access from the specified bus master are displayed on the trace panel.

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for selecting the Bus master of data access, the [Bus master of data access] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups] property is not displayed. In this case, the bus master is fixed to [CPU].

(7) [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.

Caution 1. When [Real-time RAM Monitor] is selected in the [Usage of trace function] property, [Do not output] cannot be selected from the drop-down list.

Caution 2. If Step in is executed when [CPU execution] or [Trace output] is specified and trace data is being displayed on the Trace panel correct trace data may not always be displayed.

Caution 3. **[RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]**
When this property is changed from [Do not output] to [CPU execution] or [Trace output], the timer measurement results are initialized.

(8) [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

(9) [Output timestamp]
This property specifies whether timestamp information is added to the trace data to be collected. Specify [Yes] when you want timestamp to be output.

Caution 1. **[E20(Serial) [RX200 Series]]**
If [Branch] is specified in the [Trace data type] property, timestamp cannot be output. If you want it to be output, specify [Data access].

Caution 2. **[E20 [RX100 Series]]**
This property is fixed to [No] because these microcontrollers do not support the output of timestamp information.

(10) [Trace clock count source[MHz]]
Enter a count source for the timestamp in the range 0.0001 to 999.999.
Note that if this property is blank, the set value of the [Operating frequency [MHz]] in the [Clock] category of a [Connect Settings] tab is used in place of the count source.

(11) [Division ratio of trace clock count source] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
This property is displayed only when [Yes] is selected in the [Output timestamp] property.
Select the frequency division ratio for the timestamp count source from the drop-down list.
The following frequency division ratios are displayed in the drop-down list.

1/1	The frequency of the timestamp count source is used without change.
1/16	The frequency of the timestamp count source is multiplied by 1/16 before use.
1/256	The frequency of the timestamp count source is multiplied by 1/256 before use.
1/4096	The frequency of the timestamp count source is multiplied by 1/4096 before use.

The frequency specified in the [Trace clock count source[MHz]] property is divided by the specified value (the frequency is multiplied by 1/n) and one cycle of the obtained frequency is used as the unit for timestamp count (the frequency for count value 1).

Caution 1. This property setting cannot be changed during program execution.

Caution 2. For an microcontroller that does not have the function for dividing the timestamp frequency for tracing, the [Division ratio of trace clock count source] [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups] property is not displayed. In this case, the division ratio is fixed to [1/1].

2.13.1.3 For [EZ Emulator]

Make settings in the [Trace] category on the Property panel's [Debug Tool Settings] tab.

Figure 2.142 [Trace] Category [EZ Emulator]

Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch
Output timestamp	**Yes**
Trace clock count source[MHz]	

(1) [Usage of trace function]
Only the trace function can be used.

(2) [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, CS+ continues writing trace data over the old data (default).
Stop trace	When the trace memory is filled, CS+ stops writing trace data (but does not stop program execution).
Stop	When the trace memory is filled, CS+ stops writing trace data and stops program execution at the same time.

(3) [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.

Note 1. **[RX100, RX200 Series]]**
Trace data for [Branch + Data access] cannot be collected. Therefore, this item is not displayed in the drop-down list.

Note 2. **[RX100, 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".

(4) [Output timestamp]
This property specifies whether timestamp information is added to the trace data to be collected. Specify [Yes] when you want timestamp to be output.

Caution 1. [RX200 Series]]
If [Branch] is specified in the [Trace data type] property, timestamp cannot be output. If you want it to be output, specify [Data access].

Caution 2. [RX100 Series]]
This property is fixed to [No] because these microcontrollers do not support the output of timestamp information.

(5) [Trace clock count source[MHz]]
Enter a count source for the timestamp in the range 0.0001 to 999.999.
Note that if this property is blank, the set value of the [Operating frequency [MHz]] in the [Clock] category of a [Connect Settings] tab is used in place of the count source.

2.13.1.4 For [Simulator]

Make settings in the [Trace] category on the Property panel's [Debug Tool Settings] tab.

Figure 2.143 [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

(1) [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).

(2) [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.

(3) [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, CS+ 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, CS+ 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.

(4) [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).

(5) [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, 2M, 3M

2.13.2 Collecting an execution history up to a halt

The debug tool has a pre installed 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 "2.13.6 Displaying an execution history".

Caution [Simulator]
With string manipulation instructions and multiply-and-accumulate instructions, only a history of the first and final accesses 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 check box 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.

2.13.3.1 Setting a trace start event and a trace end event
2.13.3.2 Combining multiple events [E1] [E20] [EZ Emulator]
2.13.3.3 Executing the program
2.13.3.4 Editing trace start and trace end events
2.13.3.5 Deleting a trace start or trace end event

Caution 1. Also see "2.17.7 Points to note regarding event setting" for details on trace settings, including the allowable number of valid events.

Caution 2. [Simulator]
With string manipulation instructions and multiply-and-accumulate instructions, only a history of the first and final accesses is collected.

Caution 3. **[Simulator]**
If you perform a trace from a trace start event until a trace end event, the simulator will not include the trace end event in the results of the trace. For this reason, if you are using a simulator, set the trace end event one line below the range for which you require display of the trace data.

2.13.3.1 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.

(1) 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.

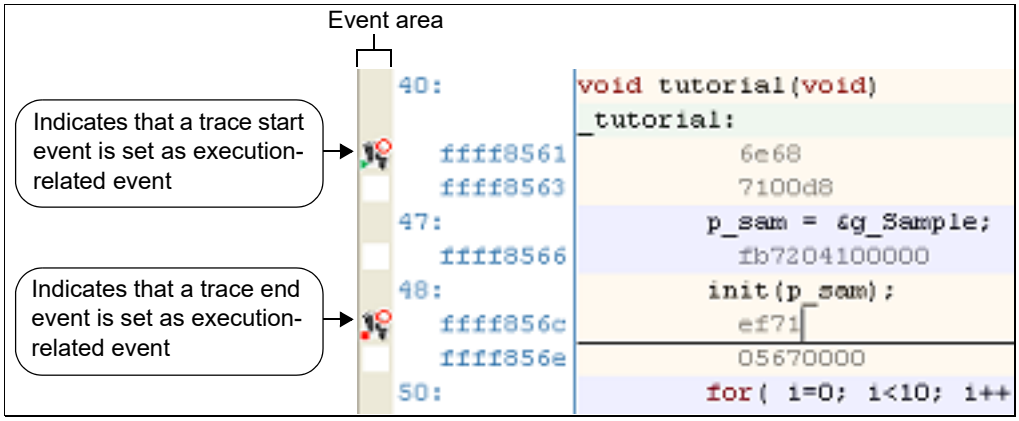
Note Trace start and trace end events cannot be set on lines that have no addresses displayed.

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 2.11 Trace Start Event and Trace End Event Marks (Execution-related Events)

Type	Event mark
Trace start	
Trace end	

Figure 2.144 Example of Trace Start and Trace End Events Set by Using Execution-related Events (For the Disassemble panel)



(2) 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 2.12 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/disassembled text is dis-

played.
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 inside a function, or static variables inside a 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	[Trace start event] Choose [Record Start R/W Value] from [Trace Settings] and then hit the [Enter] key. [Trace end event] Choose [Record End R/W Value] from [Trace Settings] and then hit the [Enter] key. If a value is specified in the text box of the menu at this time, collection of trace data is started/finished only when the specified value is used for the reading or writing. On the other hand, if no value is specified, reading or writing the selected variable by any value will start/finish the collection of trace data.

Caution 1. Only the variables in current scope are selectable.

Caution 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.

Caution 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 "2.13.3.4 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 inside a function, static variables inside a 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	[Trace start event] Choose [Record Start R/W Value] from [Trace Output] and then hit the [Enter] key. [Trace end event] Choose [Record End R/W Value] from [Trace Output] and then hit the [Enter] key. If a value is specified in the text box of the menu at this time, collection of trace data is started/finished only when the specified value is used for the reading or writing. On the other hand, if no value is specified, reading or writing the selected watch-expression by any value will start/finish the collection of trace data.

Caution 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.

Caution 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 "2.13.3.4 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.)

Caution 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. In such cases, the disassembled code displayed in the [Trace] panel will not be correct.

Caution 2. The [Trace] panel shows no trace information if the type of all trace cycles is "BCND".

Caution 3. [E20(JTAG) [RX600, RX700 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.

Remark 1. If either one of the trace start and trace end events set is Enabled, the unconditional trace event check box on the Events panel is automatically deselected, so that the trace data from the trace start event to the trace end event is collected.

Remark 2. The trace start event may be left unset if it is unnecessary. If the trace start event is not set, trace data collection starts when program execution starts. Likewise, if the trace end event is not set, trace data collection ends when program execution stops.

Remark 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.

Remark 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.

2.13.3.2 Combining multiple events [E1] [E20] [EZ Emulator]

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 2.13 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 "2.17.4.3 Editing combination conditions of events [E1] [E20] [EZ Emulator]".

Caution Also see Cautions of "(1) Editing the combination condition" for details on combination conditions.

2.13.3.3 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 "2.13.6 Displaying an execution history".

2.13.3.4 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 trace start or trace end event (execution-related or access-related) displayed in the detailed information on the trace on the Events panel that is opened, and then click [Edit Condition ...] on the context menu.
Edit the events in a dialog box that is opened by this operation.
For details on how to edit in an execution-related event dialog box, see "2.17.4.1 Editing execution-related events".
Also, for details on how to edit in an access-related event dialog box, see "2.17.4.2 Editing access-related events".

2.13.3.5 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.
You can also delete a trace start or trace end event on the Events panel which opens by selecting [View] menu >> [Event]. After selecting the trace start or trace end event (execution-related or access-related) displayed in the detailed information on the trace in the Events panel, click ( ) button on the toolbar to delete it. (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.

Caution 1. Also see "2.17.7 Points to note regarding event setting" for details on trace settings, including the allowable number of valid events.

Caution 2. [Simulator]
With string manipulation instructions and multiply-and-accumulate instructions, only the first and final data accesses are subjected to the event check.

Remark 1. Accesses by DMAC (Direct Memory Access Controller) and DTC (Data Transfer Controller) are not supported.

Remark 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.

Remark 3. When a point trace event and trace start and end events are specified together, the operation differs depending on the debug tool used.

- **[E1] [E20] [EZ Emulator]**
  Only the data that matches the point trace event condition within the section between the specified trace start and end events is collected as trace data.
- **[Simulator]**
  The execution history within the section between the specified trace start and end events and the data that matches the point trace event condition regardless of the section between the specified trace start and end events are collected as trace data.

(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/disassembled 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 inside a function, or static variables inside a 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].

Access Type	Operation
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 inside a function, static variables inside a 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 "2.13.6 Displaying an execution history".
When editing a trace event (access-related) you have set as a point trace event, first open the Events panel by selecting [View] menu >> [Event]. After selecting the trace event (access-related) displayed in the detailed information on the point trace event, click [Edit Condition ...] on the context menu. (See "2.17.4.2 Editing access-related events".)
When deleting a trace event (access-related) you have set as a point trace event, first open the Events panel by selecting [View] menu >> [Event]. After selecting the trace event (access-related) you wish to delete in the detailed information on the point trace event in the Events panel, click ( ) button on the toolbar to delete it. (See "2.17 Event Management".)

2.13.5 Stopping/Restarting Collection of Execution History [E20] [Simulator]

It is possible to temporarily stop or restart the collection of execution history during program execution.

Caution 1. If the trace function is halted or restarted during program execution, realtime capability may be impaired.

Caution 2. This facility cannot be used when a trace start or trace end event is set.

Caution 3. [E1] [E20 [RX200 Series]] [EZ Emulator]
The facility to stop or restart the trace function during program execution is not supported.

Caution 4. [E20(JTAG) [RX600, RX700 Series]]
If [Do not output] is selected in the [External trace output] property in the [Trace] category on the Property panel's [Debug Tool Settings] tab, this facility cannot be used. To use it, select [CPU execution] or [Trace output].

Caution 5. [E20(JTAG) [RX600, RX700 Series]]
If [Real-time RAM monitor] is selected in the [Usage of trace function] property in the [Trace] category on the Property panel's [Debug Tool Settings] tab, this facility cannot be used. To use it, select [Trace].

Caution 6. This facility cannot be used when an action event is set.

2.13.5.1 To temporarily stop collection of execution history

By selecting [Stop Trace] from the context menu on the Trace panel during program execution, it is possible to temporarily stop collection of trace data.

Use this feature when you want to stop only the trace function without halting the program and check the trace data that has been collected until you stop it.

2.13.5.2 To restart collection of execution history

If you've halted the trace function during program execution, you can start collection of trace data again by selecting [Start Trace] from the context menu on the Trace panel.

Note that the trace data on the Trace panel that has been collected before you restart is cleared once.

2.13.6 Displaying an execution history

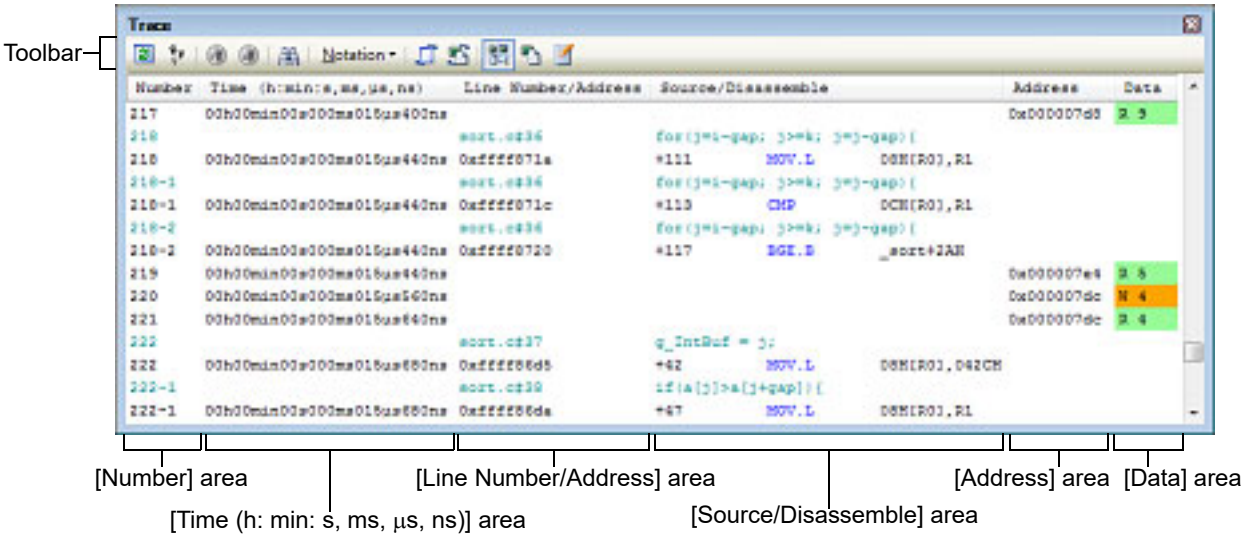
To display the collected trace data, use the Trace panel shown below.

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 2.145 Displaying Trace Data (Trace Panel)



This section describes the following.

2.13.6.1 Changing the display mode
2.13.6.2 Changing the form in which values are displayed
2.13.6.3 Getting linked to other panels

2.13.6.1 Changing the display mode

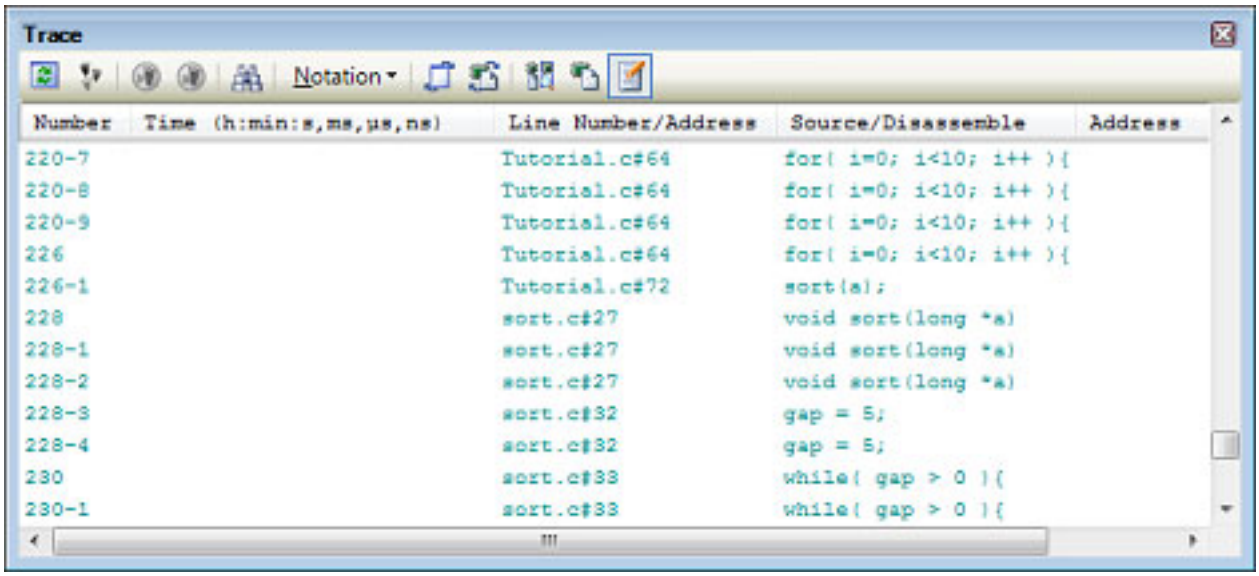
By clicking one of the toolbar buttons shown below, it is possible to change the display mode as suitable for the purpose of use.

However, these buttons are disabled while the trace function is in operation.

Table 2.14 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 2.146 Example of Source Display Mode (Trace Panel)



2.13.6.2 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.

2.13.6.3 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).

2.13.7 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.

2.13.8 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.)

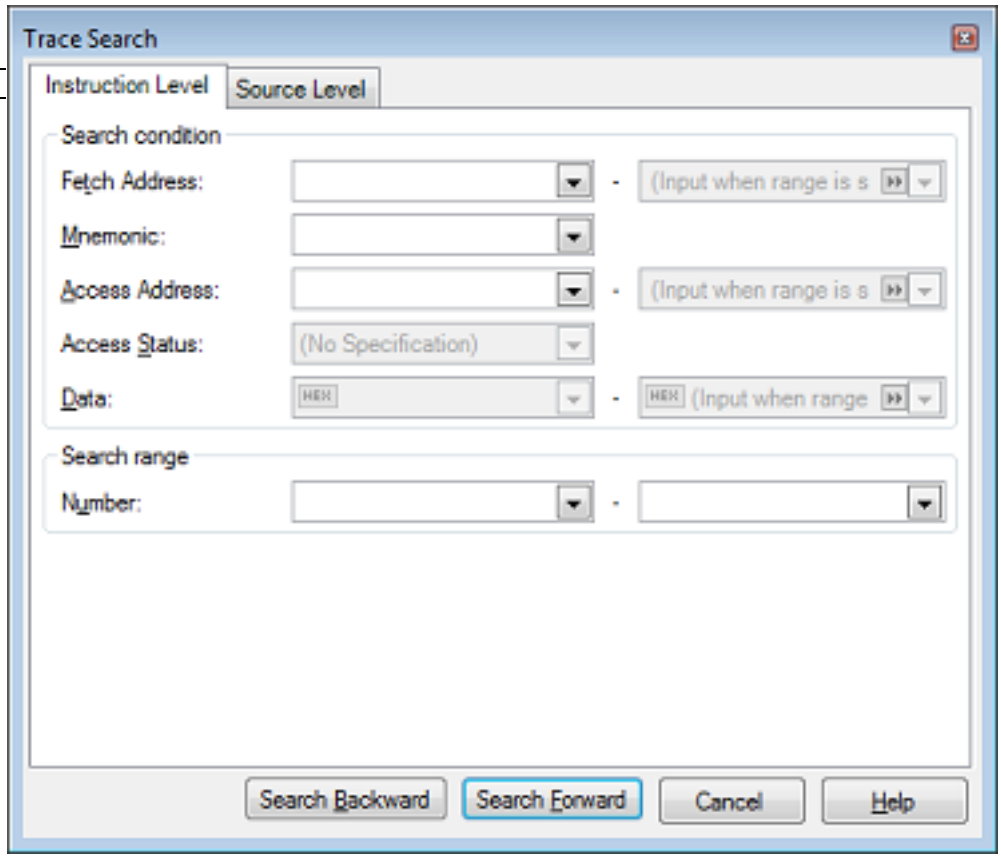
In this dialog box, perform the following operation.

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 2.147 Searching for Trace Data (Trace Search Dialog Box)



This section describes the following.

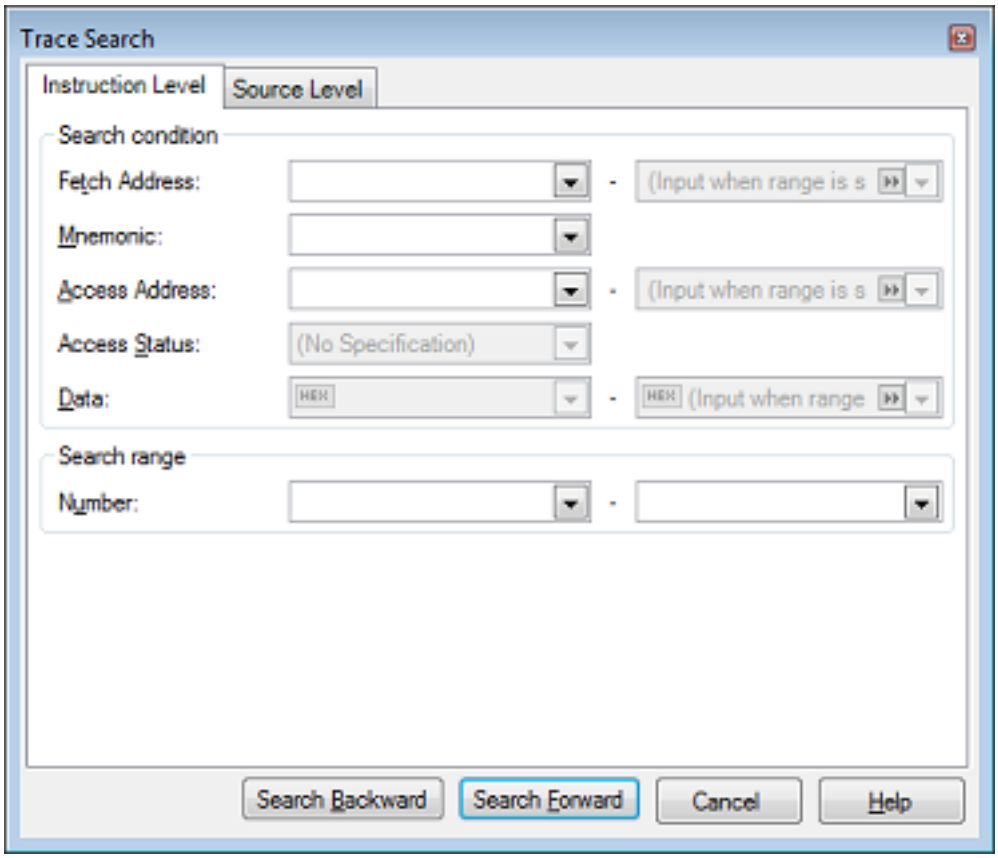
2.13.8.1 Searching at the instruction level
2.13.8.2 Searching at the source level

2.13.8.1 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 2.148 Searching for Trace Data at Instruction Level



(1) 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.

(2) 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.

(3) 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.

(4) 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 [E1] [E20] [EZ Emulator]
The types of access for which data can be collected on the Trace panel are only Read and Write. Therefore, do not select Read/Write, Vector Read or DMA from the drop-down list.
[Simulator]
The types of access for which data can be collected on the Trace panel are only Read, Write, and Vector Read. Therefore, do not select Read/Write or DMA from the drop-down list.

(5) 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.

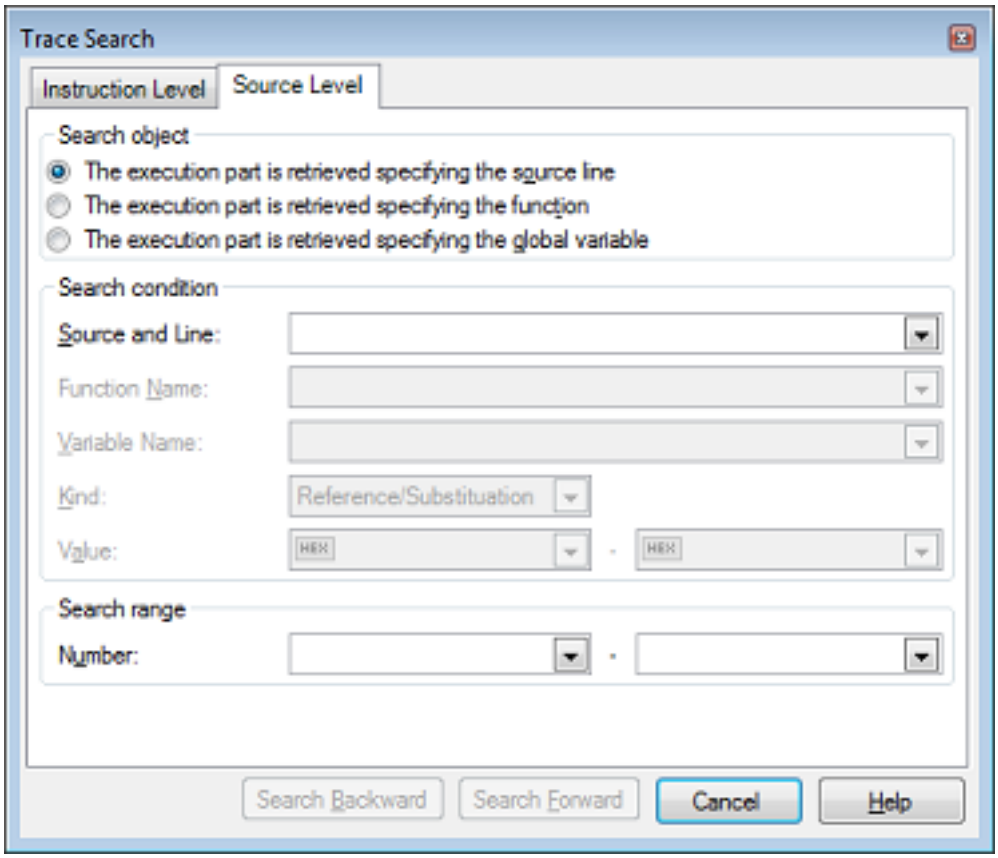
(6) Specifying the [Number]
Specify a range of trace data to search by numbers displayed in the [Number] area of the Trace panel.
Specify start and end numbers in the left and right text boxes, respectively. (By default, "0" to "*last number*" are specified.)
Enter a number in decimal notation directly 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.

(7) 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.

2.13.8.2 Searching at the source level

Search for trace data at the source level.
Select the [Source Level] tab.

Figure 2.149 Search for Trace Data at Source Level



(1) 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.

(a) 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.

Example 1. main.c#40

Example 2. main.c

Example 3. main

(b) 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.

(c) 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) 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.

(a) 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.

(b) 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.

(c) 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.

(3) 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.

(a) 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.

(b) Specifying the [Kind]
Select the type of access ([Reference/Substitution] (default), [Reference], or [Substitution]) from the drop-down list.

(c) 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.

(d) 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.

(e) 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.

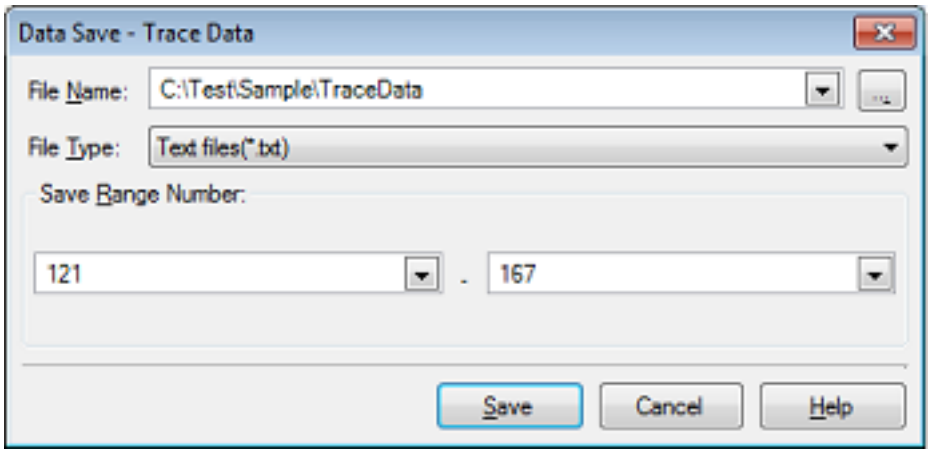
2.13.9 Saving the displayed content of an execution history

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 2.150 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 2.151 Trace Data Output Image When Saved

```
Number     Time     Line Number/Address     Source/Disassemble     Address    Data
-------------------------------------------------------------------------------
Number     Time     Line Number/Address     Source/Disassemble     Address    Data
   :         :               :                       :                 :         :
```

2.14 Measuring the Execution Time

This section describes how to measure a program's execution time.

Caution 1. [EZ Emulator [RX200 Series]]
If the reset signal is input through the reset pin, execution time that has been measured is cleared when the EZ Emulator restarts the execution of the program.

Caution 2. [E1/E20/EZ Emulator [RX600 Series]]
The number of times interrupts or exceptions occur and the number of RTE or RTFI instructions must be the same, so obtaining this balance is a precondition. If the number of times interrupts or exceptions occur and the number of RTE or RTFI instructions are not the same, that is, balance has not been obtained, correct measurement of the number of cycles for processing is not possible. Also, data can only be retained for 16 levels of nesting. Therefore, if nesting of interrupts or exceptions reaches the 17th level, correct measurement is not possible.

2.14.1 Setting the timer measurement operation [E1] [E20] [EZ Emulator]

Using timer measurements requires that you make settings related to timer measurements via the [Operating frequency[MHz] [E1][E20] [EZ Emulator]] property under the [Clock] category on the [Connect Settings] tab page and the [Timer] [E1] [E20] [EZ Emulator] category on the [Debug Tool Settings] tab page of the Property panel.

Caution [RX100 Series other than RX140 Group]
Products in this series do not support timer measurement.

Figure 2.152 [Clock] Category

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Operating frequency[MHz]	25.0000
Allow changing of the clock source on writing internal flash memory	No

Figure 2.153 [Timer] Category [RX600, RX700 Series]

Timer	
Use 64bit counter	No

(1) [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.

Caution 1. The result of timer measurement event is calculated based on the operating frequency entered in this property and count values. Therefore, using a program that switches operating frequencies while the program is running will result in incorrect measurement result.

Caution 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) [Use 64bit counter] [RX600, RX700 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, RX140 Group]
This property is not displayed because its measurement counter is 24 bits x 1 only.

2.14.2 Measuring execution time from start to stop

The debug tool has a pre installed 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.

2.14.2.1 Checking the status bar for confirmation
2.14.2.2 Checking the Events panel for confirmation

Caution 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.

Caution 2. The execution time displayed in the Status bar and the Events panel is the time measured while the program is being executed. The execution time cannot be measured when Step In, Step Over or Return Out is performed.
[E1] [E20]
Values smaller than 100 μs are discarded.

Caution 3. The EZ Emulator does not support the Run-Break timer function.

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).

2.14.2.1 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").

Remark When the EZ Emulator is in use, the status shown in the status bar is always "Not measured" because the EZ Emulator does not support the Run-Break timer function.

Figure 2.154 Example of a Run-Break Timer Event Measurement Result (Status Bar)

2.14.2.2 Checking the Events 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 2.155 Example of a Run-Break Timer Event Measurement Result (Events Panel) [E1] [E20]

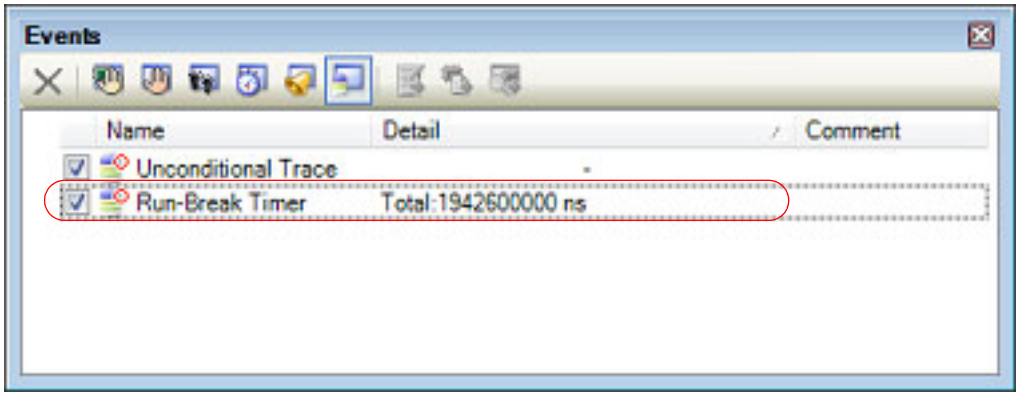
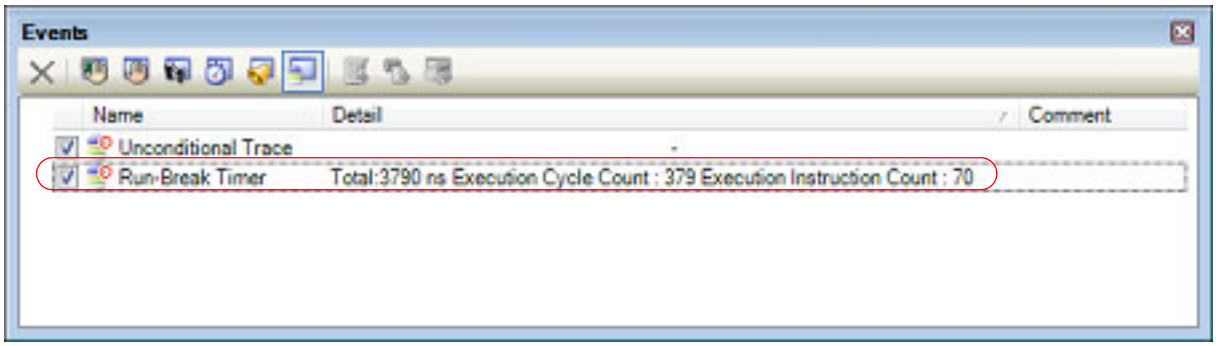


Figure 2.156 Example of a Run-Break Timer Event Measurement Result (Events Panel) [Simulator]



2.14.3 Measuring execution time in a section

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.

2.14.3.1 Setting the timer start event and timer end event
2.14.3.2 Execute the program
2.14.3.3 Editing a timer measurement event [E1] [E20] [EZ Emulator]
2.14.3.4 Editing timer start and timer end events
2.14.3.5 Deleting a timer start event or timer end event

Caution 1. Also see "2.17.7 Points to note regarding event setting" for details on timer measurement event settings, including the allowable number of valid events.

Caution 2. [E1(Serial)/E20(Serial)/EZ Emulator [RX200 Series]]
The timestamp information on the Trace panel is realized using the timer measurement counter. Therefore, if a timer measurement event is set on the Events panel, expected timestamp information is not displayed.

Caution 3. [E1] [E20] [EZ Emulator]
When a timer start or end event is added or deleted or a timer measurement event is edited, the result of the next measurement will not be accumulated.
[Simulator]
When a timer start or end event is added, deleted, or edited, the result of the next measurement will not be accumulated.

2.14.3.1 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.

(1) How to set a timer start event

- To set from the Editor panel/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] from the 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] from the 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] from the 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.

(2) How to set a timer end event

- To set from the Editor panel/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] from the context menu and from it, choose [Stop 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] from the 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] from the 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.

Note 1. Timer start event/timer end event cannot be set to lines with no address indication.

Note 2. **[E1] [E20] [EZ Emulator]**
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] [EZ Emulator] category on the Property panel's [Debug Tool Settings] tab is set, as shown below.

Microcontroller	[Use 64bit counter] property	Function
RX600, RX700 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

[Simulator]
Only one section is available and desired timer channels cannot be specified; the [Set Timer *] menu is not displayed.

Note 3. This watch-expression can only be a global variable, static variable inside a function, static variable inside a file, and an IO register.

Caution 1. [E1] [E20] [EZ Emulator]
Timer measurement is possible even if either a timer start or timer end event is set. When only a timer start event is set, measurement ends as soon as the program stops running. When only a timer end event is set, measurement starts as soon as the program starts running.
[Simulator]
Timer measurement works only when both timer start and end events are specified. If only one of the start and end events is specified, execution time cannot be measured.

Caution 2. **[Simulator]**
Only a single event can be specified for each of timer start and end events. Multiple events cannot be specified in combination.

Figure 2.157 Example of Setting a Timer Start Event (Access-related) on a Variable in the Source Text [E1] [E20] [EZ Emulator]

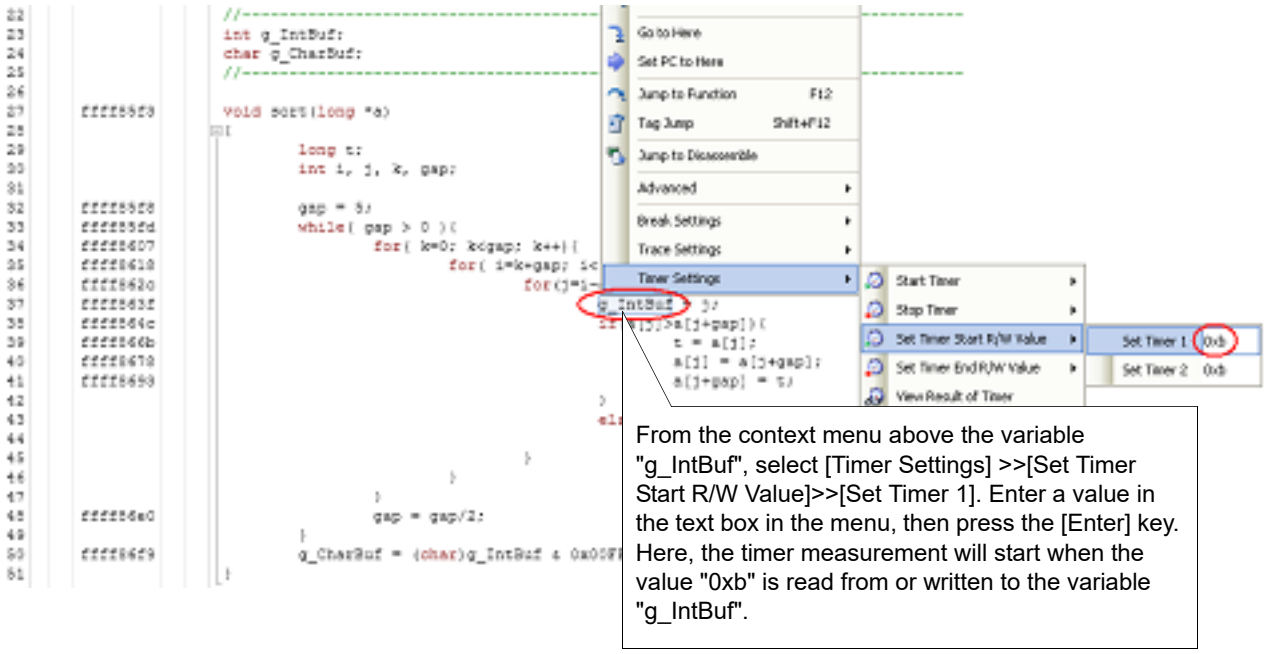


Figure 2.158 Example of Setting a Timer Start Event (Access-related) on a Variable in the Source Text [Simulator]

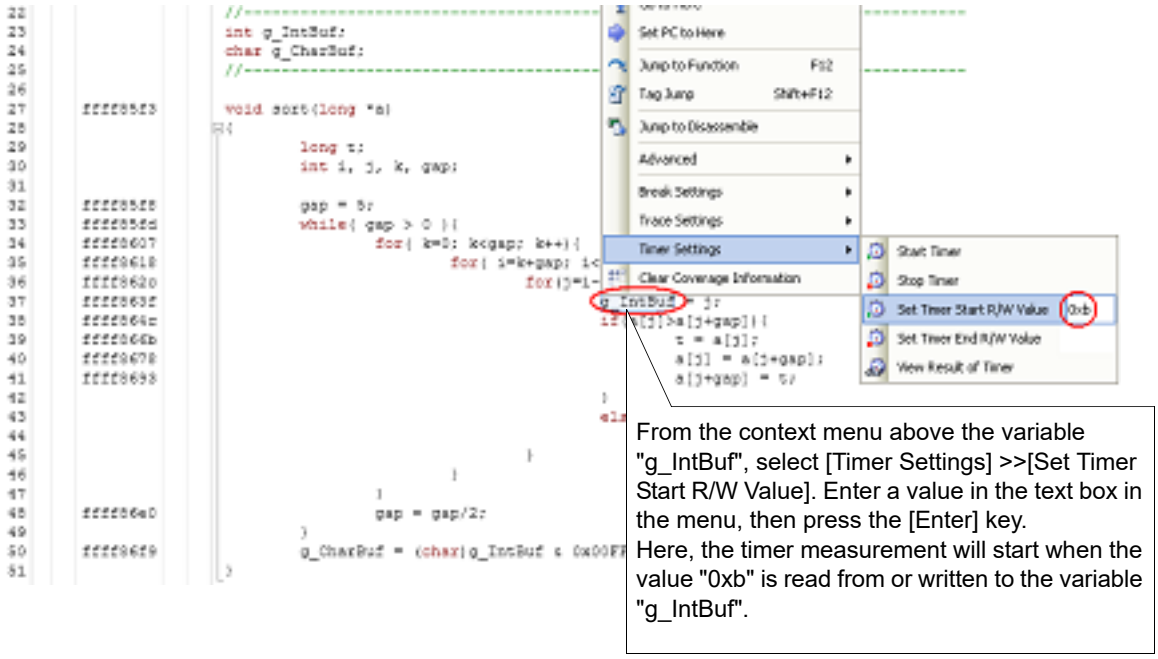


Figure 2.159 Example of Setting a Timer Start Event (Access-related) on a Watch-expression [E1] [E20] [EZ Emulator]

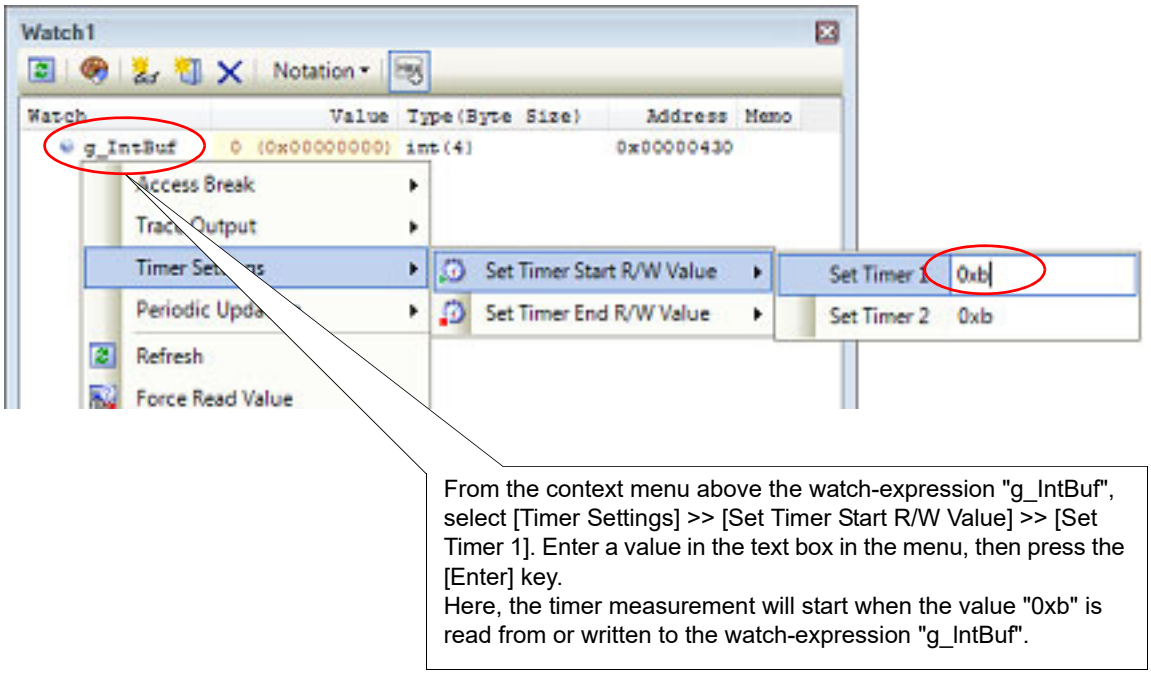
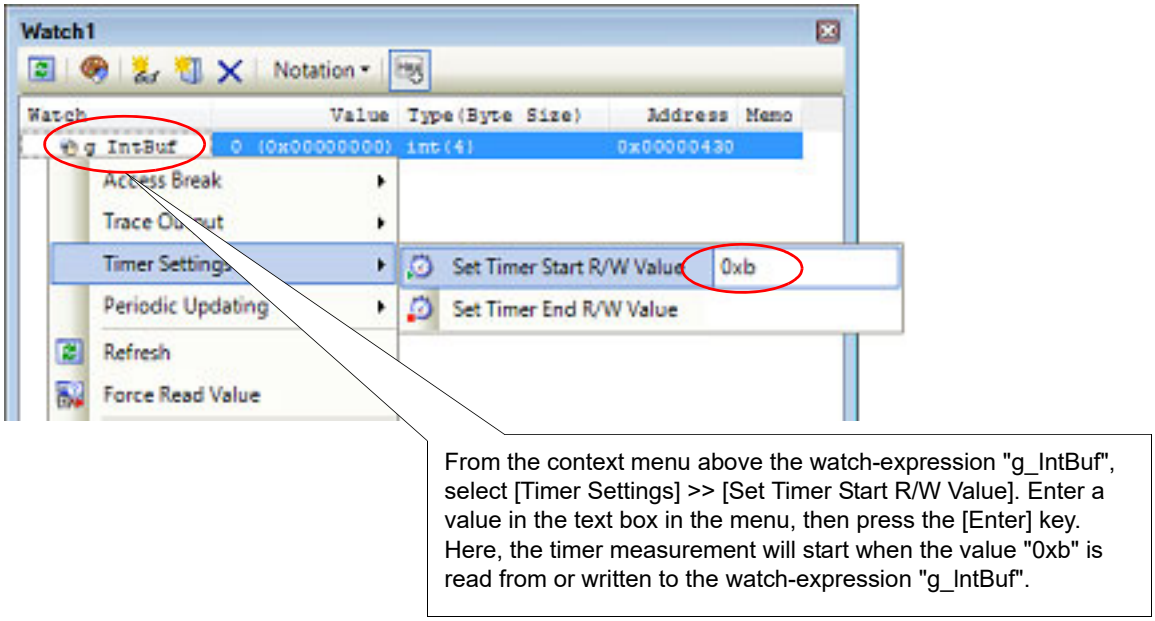


Figure 2.160 Example of Setting a Timer Start Event (Access-related) on a Watch-expression [Simulator]



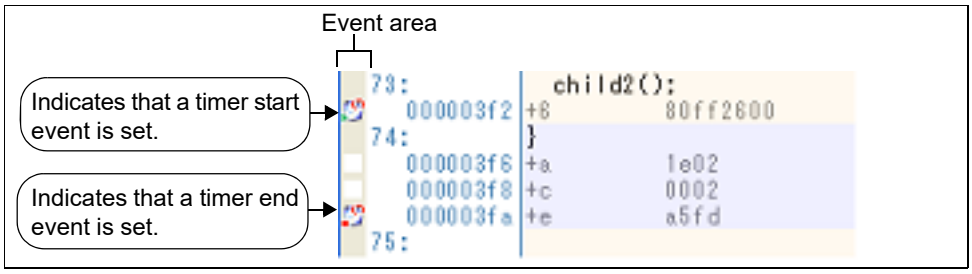
When timer start and end events are set, the following event marks are displayed in the event area of the relevant line, address, or watch-expression.

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 2.15 Event Marks of the Timer Start and End Events

Type	Event mark
Timer start	
Timer end	

Figure 2.161 Example of Timer Start and End Events Set (Disassemble panel)



Caution [Simulator]
If the timer start and end events occur at the same instruction execution, the operation depends on the measurement state as follows.

- When timer measurement has started:
  Timer measurement stops and the measured time is reflected in the result.
- When timer measurement has not started:
  Timer measurement is not done.

Remark Event marks differ depending on the event state (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.

2.14.3.2 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 2.162 Example of Measurement Results by Timer Measurement Event (Timer Start and End Events) [E1] [E20]

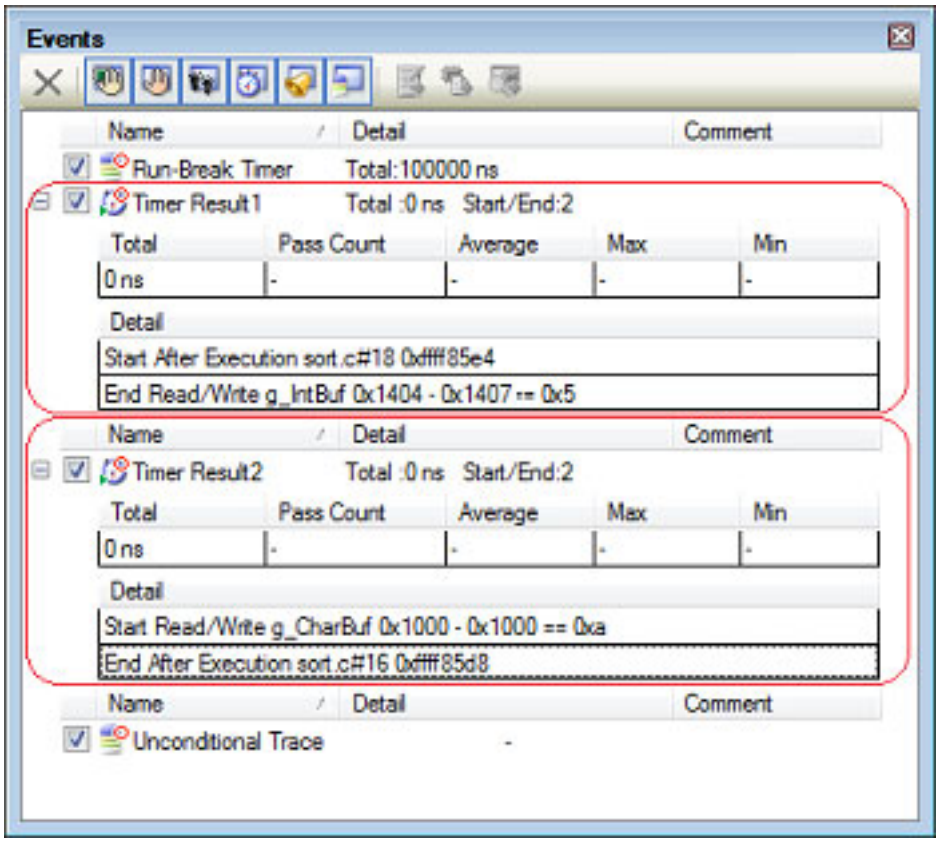


Figure 2.163 Example of Measurement Results by Timer Measurement Event (Timer Start and End Events) [EZ Emulator]

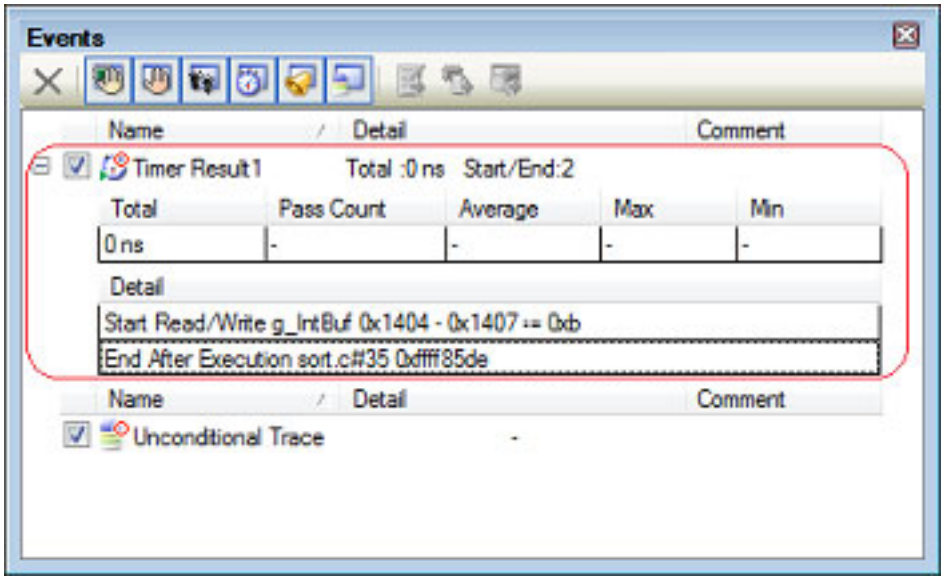
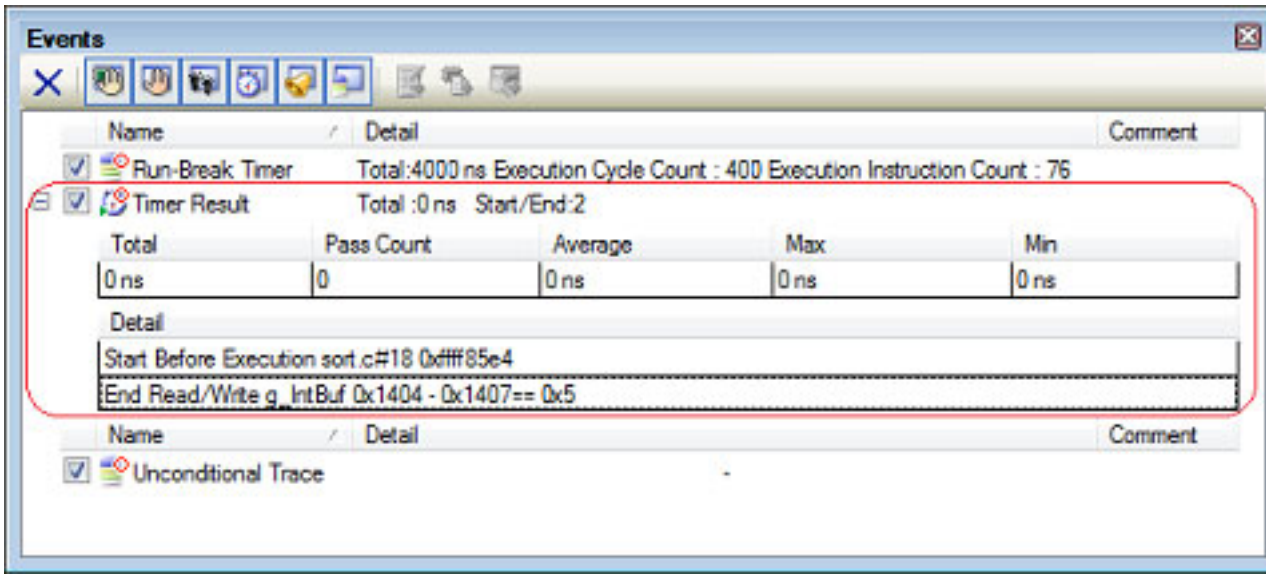


Figure 2.164 Example of a Measurement Result by Timer Measurement Event (Timer Start and End Events) [Simulator]



2.14.3.3 Editing a timer measurement event [E1] [E20] [EZ Emulator]

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] [EZ Emulator]. This dialog box is opened by selecting the timer measurement event you wish to edit on the Events panel then selecting [Edit Condition ...] from the context menu.

(1) 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, RX140 Group]
Only [Execution cycle] is displayed for the measurement item.

Remark 1. The total execution time is measured when [Execution cycle], [Exception and interrupt cycle], [Exception cycle], or [Interrupt cycle] is selected.

Remark 2. The number of passes (pass count) is measured when [Execution count], [Exception and interrupt count], [Exception count], or [Interrupt count] is selected.

Remark 3. Measurement of average, maximum, and minimum execution times is not available.

(2) To edit the [Execution only once] property
When you specify [Yes] for the [Execution only once] 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, RX700 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

Caution **[RX600, RX700 Series]**
If you specify [Yes], the timer measurement will be suspended when the timer start event occurs twice even though the timer end event has not occurred.

2.14.3.4 Editing timer start and timer end events

To edit a timer start or timer end event you've set, choose [Event] from the [View] menu, select the timer start or timer end event (execution-related or access-related) displayed in the detailed information on the timer result on the Events panel that is opened, and then click [Edit Condition ...] on the context menu.
Perform the 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 "2.17.4.1 Editing execution-related events".
Also, for details on how to edit in an access-related event dialog box, see "2.17.4.2 Editing access-related events".

2.14.3.5 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.
You can also delete a timer start or timer end event on the Events panel which opens by selecting [View] menu >> [Event]. After selecting the timer start or timer end event (execution-related or access-related) displayed in the detailed information on the timer result in the Events panel, click ( ) button on the toolbar to delete it (see "2.17.5 Deleting events").

Caution If either a timer start or timer end event is deleted from the event marks on the event area, all of the corresponding event marks are deleted.

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"), as shown below.

Table 2.16 Range of Measurable Time

Debug tool	Run-Break timer event		Timer measurement event	
E1(Serial)/E1(JTAG)/ E20(Serial)/E20(JTAG) [RX600, RX700 Series]	Min	100 microseconds	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(Serial)/E1(JTAG)/ E20(Serial)/E20(JTAG) [RX200 Series, RX140 Group]	Min	100 microseconds	Min	-
	Max	Approx. 72 hours	Max	Depends on CPU clock frequency 24-bit counter x 1 channel Overflow detection available

Debug tool	Run-Break timer event	Timer measurement event
Simulator	Depends on CPU clock frequency 64-bit counter Overflow detection available	Depends on CPU clock frequency 64-bit counter Overflow detection available

2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]

This section describes coverage measurements that are conducted using the coverage facility.

There are several kinds of coverage measurement methods. Of these, CS+ 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 CS+ performs coverage measurements are as follows:

Table 2.17 Subject Areas of Coverage Measurements

Debug Tool	Subject area
Simulator	Internal ROM/RAM, emulation ROM/RAM
E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]	Any desired area (up to four areas in 4-Mbyte units)

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%.

Caution 1. In the E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups], this function cannot be used together with the trace function or real-time RAM monitor function.

Caution 2. In the E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups], data coverage cannot be measured.

Caution 3. In the E20 (JTAG) [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Group], if you connect the emulator by using hot plug-in, the coverage measurement function is not available.
If you use hot plug-in to connect the emulator, please select [No] for the [Use code coverage function] property in the [Coverage] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]] category on the [Debug Tool Settings] tab on the Property panel.

Caution 4. In the E20 (JTAG) [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Group], when the error message "The system was reset." is displayed during program execution, coverage from the start of the program to the system reset will not have been measured.

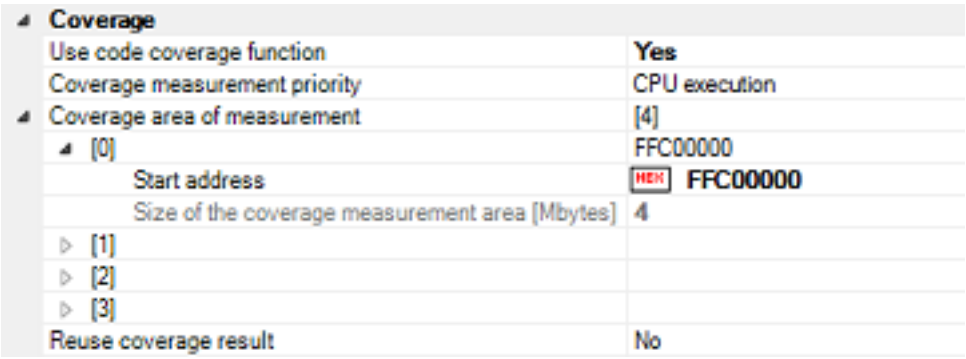
2.15.1 Configure the coverage measurement

To use the coverage facility, it is necessary to make coverage measurement-related settings in advance.

Make settings in the [Coverage] [Simulator] category, or in the [Coverage] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]] category on the [Debug Tool Settings] tab on the Property panel.

Figure 2.165 [Coverage] Category [Simulator]

Figure 2.166 [Coverage] Category [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]



(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 result]
This property is displayed only when [Use coverage function] property is set to [Yes].
Specify from the drop-down list whether to automatically save the acquired code coverage measurement result when disconnecting from the debug tool 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.)

(3) [Use code coverage function]
Specify from the drop-down list whether or not you want to use the code coverage function.
To use the code coverage function, select [Yes]. (By default, [No] is selected.)
This function can be selected only while CS+ is disconnected from the debug tool.

(4) [Coverage measurement priority]
This property is displayed only when [Yes] is specified in the [Use code coverage function] property.
Select the mode for measuring coverage from the drop-down list.
The following coverage measurement priorities are displayed in the drop-down list.
- CPU execution, Coverage measurement

When [CPU execution] is selected, the coverage data may be lost during data output.
When [Coverage measurement] is selected, CPU execution may stop to output coverage data, affecting real-time performance of program execution.

(5) [Coverage area of measurement]
This property is displayed only when [Yes] is specified in the [Use code coverage function] property.
Specify the ranges where you want to measure coverage.
Up to four ranges can be specified in 4-Mbyte units for coverage measurement.
Specify a multiple of 4 Mbytes as the start address (if the specified value is not a multiple of 4 Mbytes, the value is automatically corrected)

(6) [Reuse coverage result]
This property is displayed only when [Yes] is specified in the [Use code coverage function] property.
Specify from the drop-down list whether to automatically save the acquired code coverage measurement result when disconnecting from the debug tool and reproduce it next time you connect to the debug tool.

2.15.2 Display coverage measurement results

Coverage measurement starts (ends) automatically with the start (end) of the program execution.

(1) Code coverage rates

(a) Display of code coverage rates for source lines and disassemble lines
The code coverage rates are indicated on the Editor panel or Disassemble panel that is displaying the target program.
On each panel, the target source text lines and disassembled result lines are shown in color-coded background (see Table 2.19) according to their code coverage rate that was calculated based on the formula described in Table 2.18.
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 2.18 Calculating Code Coverage Rates 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 disassembled result line / Total number of bytes present in the address area corresponding to the disassembled result line

Table 2.19 Color-coded Code Coverage Measurement Results (Default)

Code Coverage	Background Color
100 %	Source text/ disassembled results
1 to 99 %	Source text/ disassembled results
0 % (not executed)	Source text/ disassembled results

Remark 1. The code coverage measurement result displayed on each panel is updated at every program break.

Remark 2. The above color coding rule is determined according to the settings in [General - Font and Color] category in the Option dialog box.

Remark 3. The above color coding rule does not apply to the lines that are outside the target area (see "Table 2.17 Subject Areas of Coverage Measurements").

Remark 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 2.167 Example of Code Coverage Measurement Results (Editor Panel)

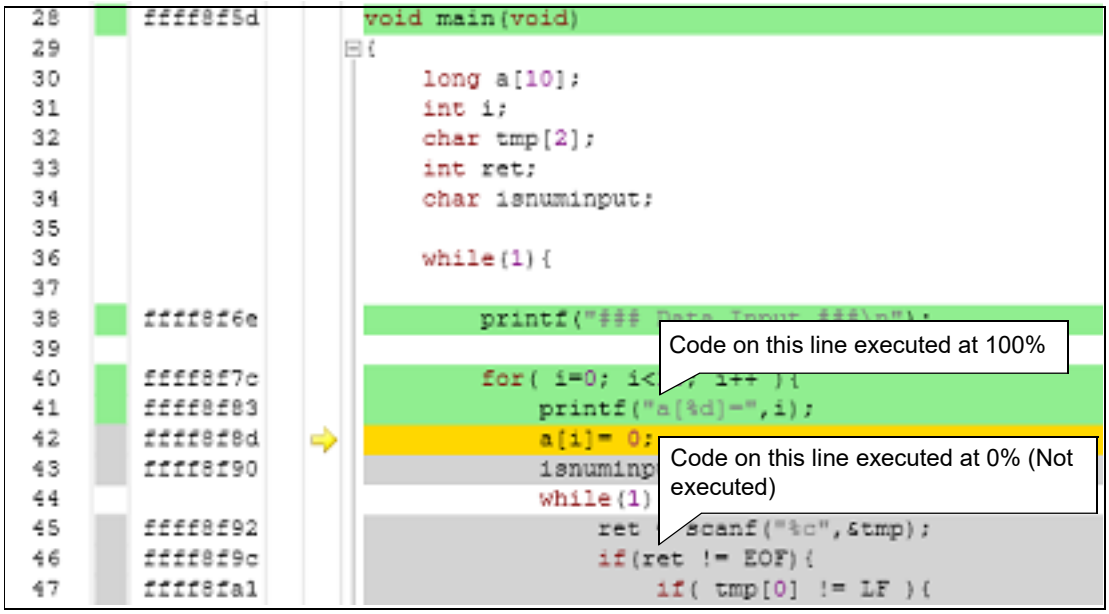


Figure 2.168 Example of the Code Coverage Measurement Result (Disassemble Panel)

```
_main:
ffff8f5d   7100d4          ADD      #-2CH,R0,R0
ffff8f60   fbc274d5ffff    MOV.L    #-00002A8CH,R12
ffff8f66   ef0b            MOV.L    R0,R11
ffff8f68   fba27bd5ffff    MOV.L
38:        printf("### Data Input ###\
ffff8f6e   fb326bd5ffff    MOV.L    #-00002AAOH,R3
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",&t
ffff8f92   7eab            PUSH.L   R11
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 List panel of the analysis tool for the code coverage rates of each function (i.e., function coverage rates).
For details on the "code coverage ratio of functions," see the separate edition "CS+ Integrated Development Environment User's Manual: Analysis Tool."

(2) Data coverage rates[Simulator]
Check the [Data Coverage[%]] item in the Variable List panel of the analysis tool for the data coverage rates of each variable.
For details on the "data coverage ratio of variables," see the separate edition "CS+ Integrated Development Environment User's Manual: Analysis Tool."

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.

2.16.1.1 Set a Printf event
2.16.1.2 Execute the program
2.16.1.3 Check the output result
2.16.1.4 Edit Printf event

Caution 1. Also see "2.17.7 Points to note regarding event setting" for details on Printf event settings, including the allowable number of valid events.

Caution 2. No Printf events occur during step execution (/ /) or execution ignoring break-related events ( ).

2.16.1.1 Set a Printf event

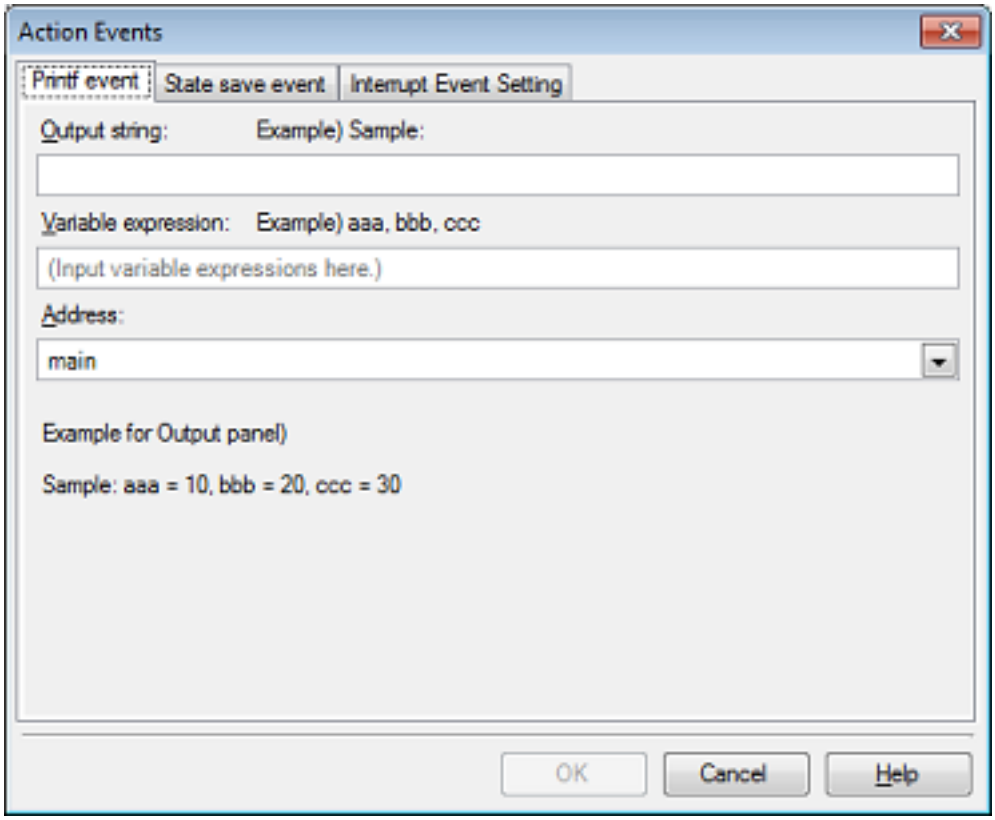
Set a Printf event to the position where you want to execute the printf command in the Editor panel/Disassemble panel.

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 2.169 Set Printf Event (Action Events Dialog Box: [Printf event] Tab)



(1) 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).

(2) 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.15 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 "2.21.2 Symbol name completion function").

(3) 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 "2.21.2 Symbol name completion function").

(4) 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").

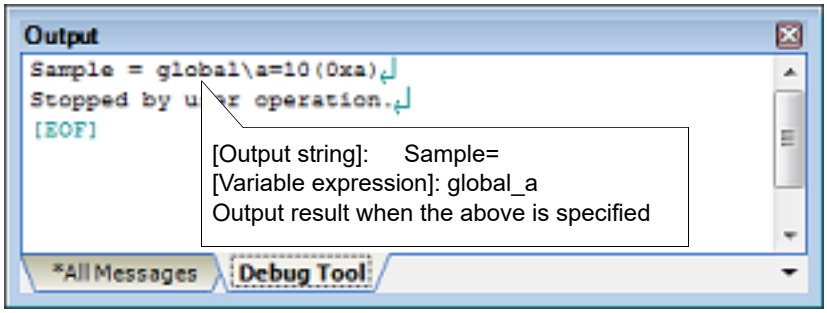
2.16.1.2 Execute the program

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.

2.16.1.3 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.28 Output Result Format of Printf Event").

Figure 2.170 Example of Output Result of Printf Event



2.16.1.4 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 Condition…] 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.

2.16.2.1 Set an interrupt event
2.16.2.2 Execute the program
2.16.2.3 Edit interrupt event

Caution Also see "2.17.7 Points to note regarding event setting" for details on interrupt event settings, including the allowable number of valid events.

2.16.2.1 Set an interrupt event

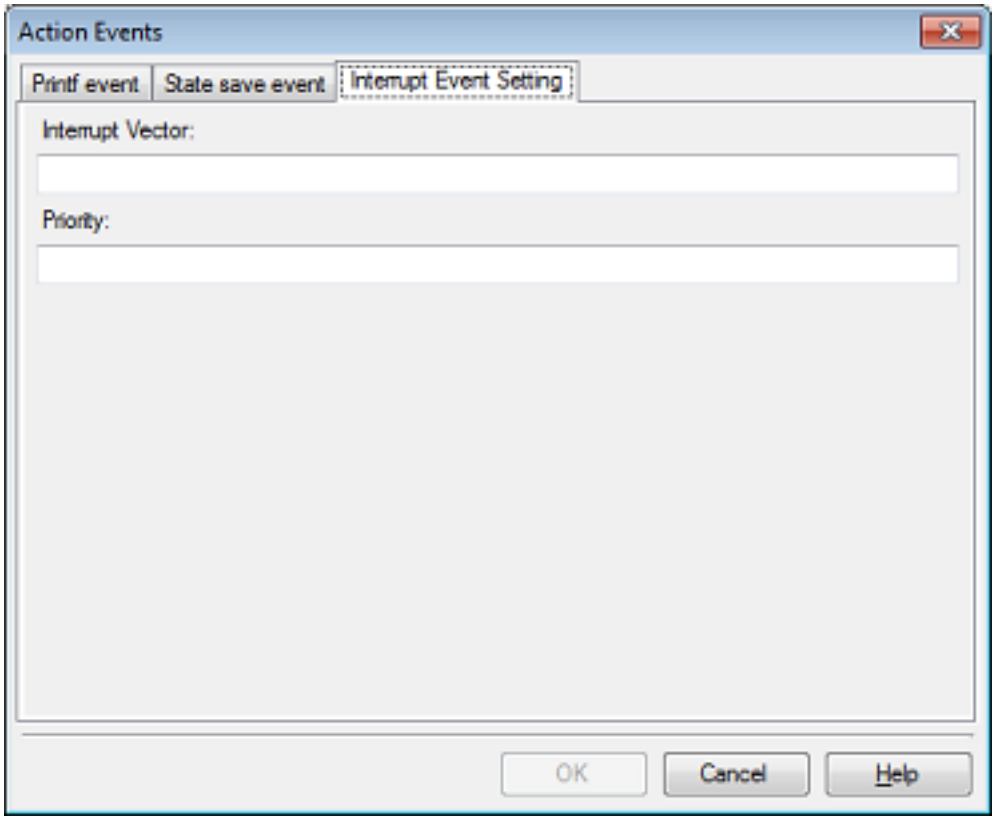
In the Editor panel/Disassemble panel, set an interrupt event at a location where you want an interrupt request to be generated.

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, then select the [Interrupt event setting] tab [Simulator].

In this dialog box, follow the steps below.

Note Interrupt events cannot be set to lines with no address indication.

Figure 2.171 Set Interrupt Event (Action Events Dialog Box: [Interrupt Event Setting] Tab



(1) Specify [Interrupt Vector]
Specify the interrupt vector by directly entering a corresponding number between 0 and 255.

(2) Specify [Priority]
Specify an interrupt's priority level.

- [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.

(3) 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").

2.16.2.2 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.

Caution 1. Once the generated interrupt request is accepted and interrupt exception executed, the interrupt request will be cleared.

Caution 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.

2.16.2.3 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 Condition…] 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].

(1) Edit [Pass Count]
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.

(2) 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 "(1) Specify [Interrupt Vector]"and" (2) Specify [Priority]").

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."

CS+ 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 Events 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.

Remark When multiple load module files are downloaded, the load module names are displayed as the detailed information on the Events panel.

Figure 2.172 Displaying the Set Events (Events Panel) [E1] [E20]

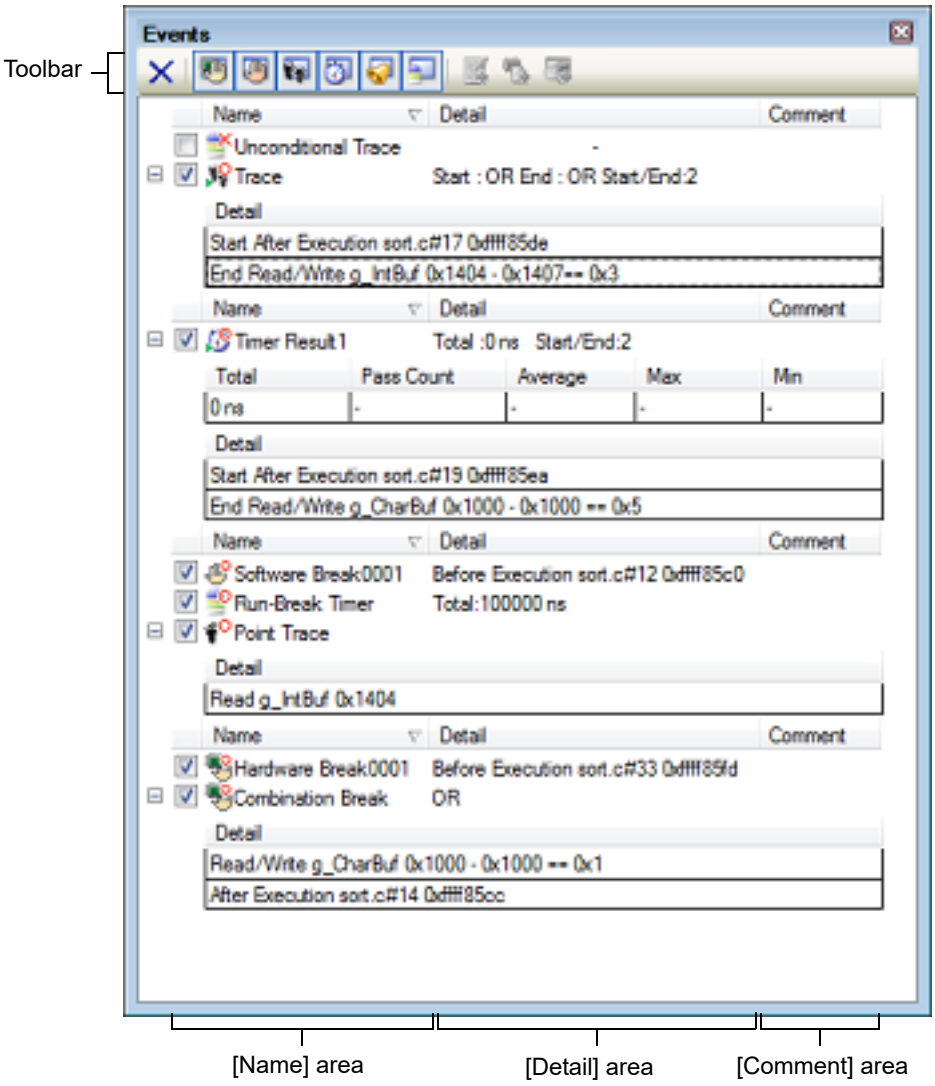


Figure 2.173 Displaying the Set Events (Events Panel) [EZ Emulator]

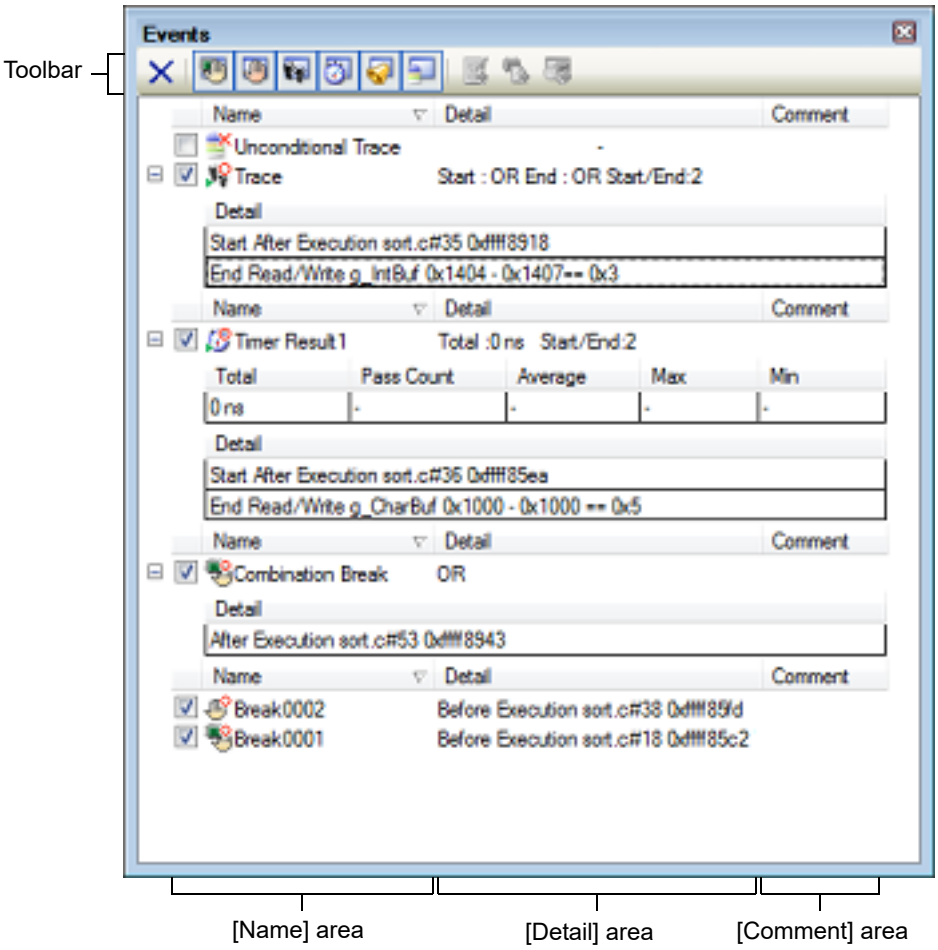
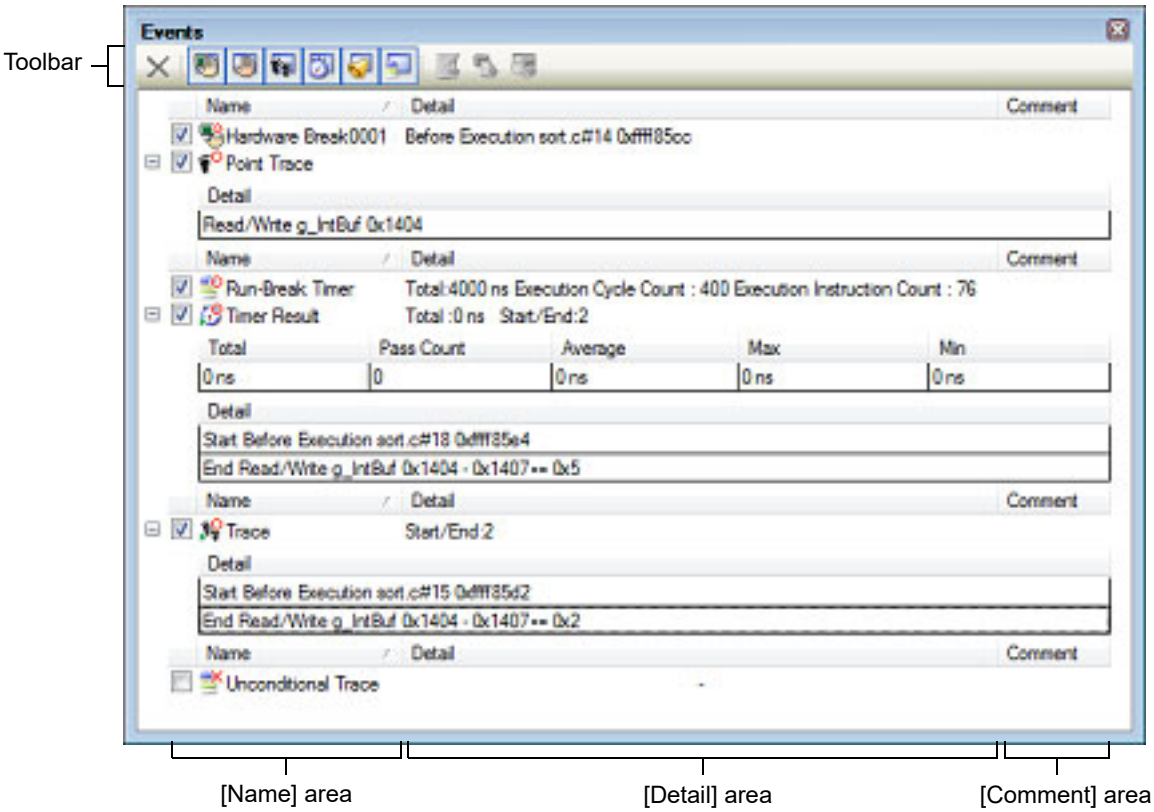


Figure 2.174 Displaying the Set Events (Events Panel) [Simulator]



2.17.1 Changing states of setting (Enabled/Disabled)

By checking or unchecking the check box 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 2.175 Event Name Check Box

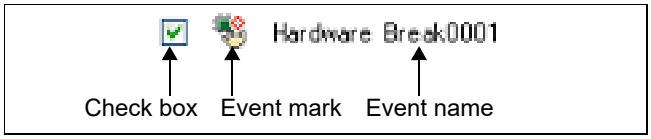


Table 2.20 Set States of Events

	Enabled	When a specified condition is met, the enabled event occurs. The event can be disabled by unchecking its check box.
	Disabled	Even when a specified condition is met, the disabled event does not occur. The event can be enabled by checking its check box.
	Pending	A specified condition cannot be set in the program to be debugged. The check box of any event in this state cannot be manipulated.

Remark 1. The Run-Break timer event (not supported by the EZ Emulator) cannot be disabled or made pending.

Remark 2. 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.

Remark 3. 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] [EZ Emulator]	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). Note

Note The EZ Emulator does not support Run-Break timer events.

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 "2.14.3.3 Editing a timer measurement event [E1] [E20] [EZ Emulator]". For information on editing of action events (Printf events and interrupt events), see "2.16 Set an Action into Programs".

2.17.4.1 Editing execution-related events
2.17.4.2 Editing access-related events
2.17.4.3 Editing combination conditions of events [E1] [E20] [EZ Emulator]

2.17.4.1 Editing execution-related events

It is possible to edit the condition of the address and 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] [EZ Emulator]**	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	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.

(1) Editing the address
Enter a hexadecimal value in the [Address] property under the [Address] category and click [OK].

(2) 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] [EZ Emulator]**
  1 to 256
- **[Simulator]**
  1 to 16,383

Caution 1. [E1] [E20] [EZ Emulator]
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.

Caution 2. [E1(Serial)/E20(Serial)/EZ Emulator [RX100, RX200 Series]]
No values other than 1 can be specified for the pass count.

2.17.4.2 Editing access-related events

The address condition and data condition for access-related events can be edited.
For details on how to edit the address and the pass count, see "(1) Editing the address", "(2) 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] [EZ Emulator]**	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	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.

(1) Editing address conditions
In the [Compare condition] property in the [Address Condition] category, specify a comparison condition from the drop-down list.
Depending on the specified comparison condition, the following address conditions can be set in combination with the address values displayed in the [Address] property in the [Address] category.

(a) When [No conditions] is specified
The address value is made the condition. When an access to the address value occurs, the condition holds true.

(b) When [Address area] is specified [E1] [E20] [EZ Emulator]
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.

Caution 1. [RX100, RX200 Series]
Comparison conditions based on an address area are not supported.

Caution 2. [RX600, RX700 Series]
You can specify "Address area" as the comparison condition only to one event.

Caution 3. [Simulator]
Address conditions cannot be specified.

(c) When [Compare with address mask] is specified [E1] [E20] [EZ Emulator]
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 comparison condition.

Caution 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

(2) Editing data conditions

(a) [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.

(b) [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.

(c) [Compare condition]
Specify a data comparison condition.
The comparison condition differs with each debug tool as follows:

[E1] [E20] [EZ Emulator]	Specified value (==)	When data comparison conditions match, the condition holds true.
	Any other value (!=)	When data comparison 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 Note	When the sign is inverted between the previously accessed data and the data accessed this time, the condition holds true.
	Difference Note	When the difference between the previously accessed data and the data accessed this time exceeds a specified value, the condition holds true.
	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.

Remark **[Simulator]**
This property is not displayed only when you've specified [No conditions] in the [Access size] property.

(d) [Compare data] [E1] [E20] [EZ Emulator]
Specify a data value with a hexadecimal number.

(e) [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<=)]

(f) [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.

(g) [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. If you specify [Yes], the data mask can 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<=)]

(h) [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.

(i) [Difference] [Simulator]
Specify comparison data with a hexadecimal number.
This property is displayed when you've specified [Difference] in the [Compare condition] property.

(j) [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 Bus master [E1/E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]

(a) [Bus master]
Specify one of the following items for the bus master.

CPU	When the specified data access from the CPU occurs, the condition holds true.
DMAC/DTC	When the specified data access from the DMAC/DTC occurs, the condition holds true.

2.17.4.3 Editing combination conditions of events [E1] [E20] [EZ Emulator]

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 Combination Condition dialog box [E1] [E20] [EZ Emulator] that is displayed by the following operation on the Events panel.

Caution [EZ Emulator]
If the reset signal is input through the reset pin, information of events that have been encountered up to that point is cleared when the EZ Emulator restarts the execution of the program.

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.

(1) Editing the combination condition
Select the combination condition between [OR], [AND] or [Sequential] from the pull-down menu. For the explanation of operation per combination condition, see "2.10.5 Set multiple break events in combination (Combination break) [E1] [E20] [EZ Emulator]", and "2.13.3.2 Combining multiple events [E1] [E20] [EZ Emulator]".

Figure 2.176 [Combination] item

Caution 1. For trace events, the combination condition available for a trace end event is only [OR].

Caution 2. [AND] cannot be used as the combination condition for both the combination break and trace start event. Similarly, [Sequential] cannot be used as the combination condition for both the combination break and trace start event.
When [AND] and [Sequential] are selected as the combination condition for either of the combination break or trace start event, only [OR] can be selected for the other.

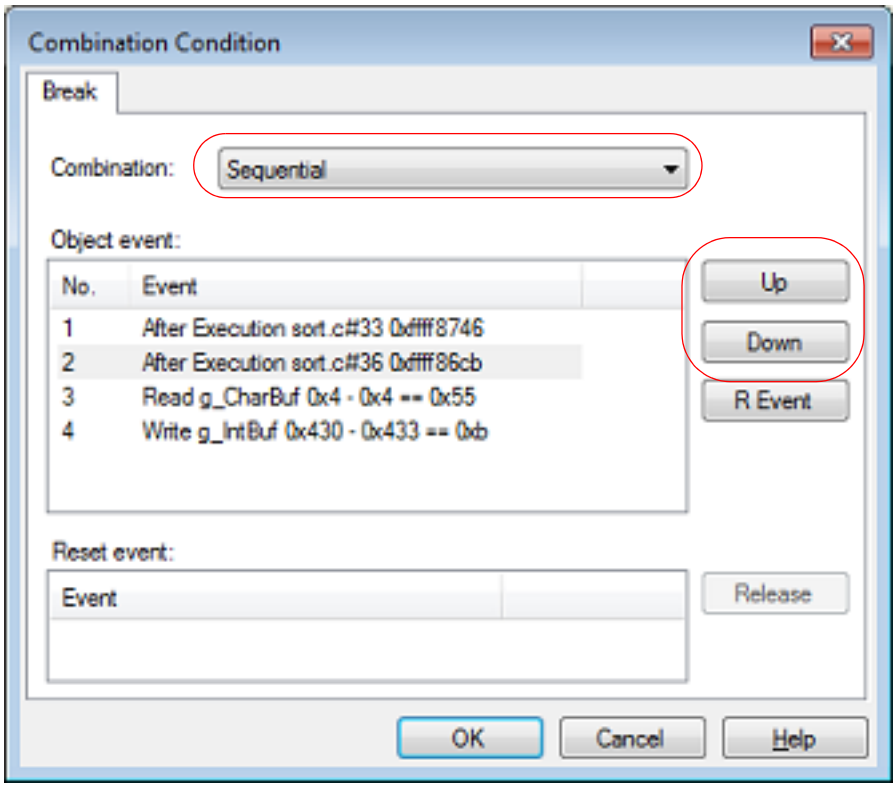
Caution 3. [RX600, RX700 Series]
When [Sequential] is selected as a combination condition, access-related events are only usable as the reset event or as the first to third events.

Caution 4. [RX600, RX700 Series]
When [Sequential] is selected as a combination condition, access-related events for which an address area condition has been set can be specified only for the 1st position.

Caution 5. [RX100, RX200 Series]
When [Sequential] is selected as a combination condition, access-related events are only usable as the first to second events.

(2) Editing the subject event
Change the order of events in the Object 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 Object event list.

Figure 2.177 Changing the Order of Events in the Object Event List (Sequential)



(3) 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 Object 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

Caution 1. Only one reset event is specifiable.
If you click the [R Event] button with the reset event registered, the event selected in the Object event list and the registered reset event will replace each other.

Caution 2. If you change the combination condition from [Sequential] to [OR] or [AND] with the reset event registered, the event in the Reset event list cannot be included in the Object event list. In order to include it in the Object event list, you need to first cancel the reset event registration before making changes to the combination conditions.

Caution 3. When [Sequential] is selected as a combination condition, the pass count cannot be specified for an event that has been registered as the reset event.

Caution 4. [RX600, RX700 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 2.178 Registering a Reset Event (Sequential)

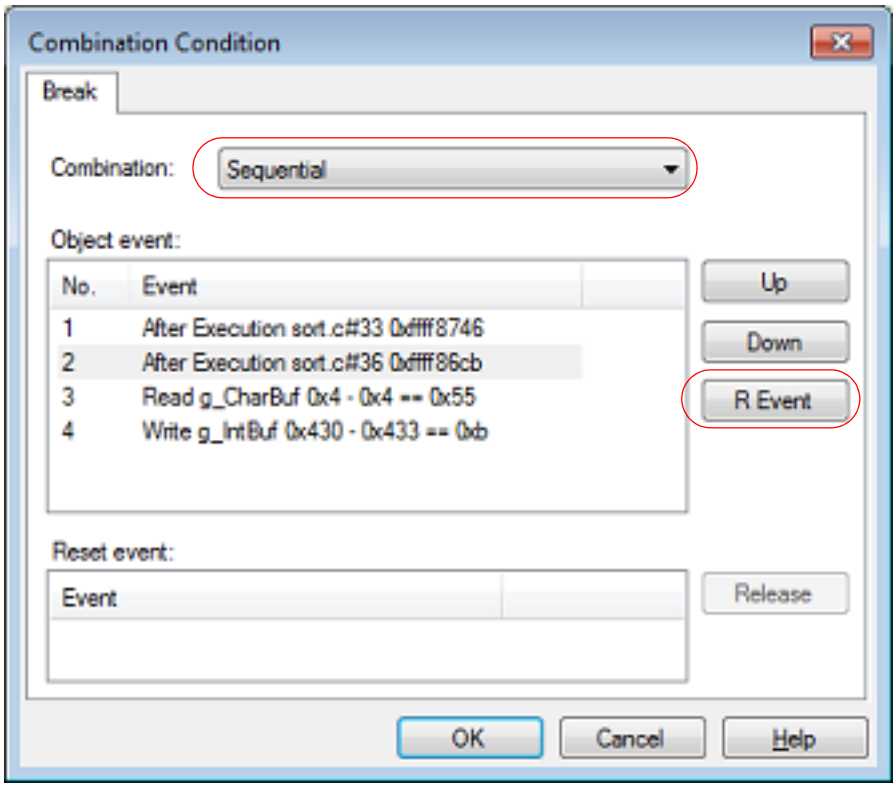
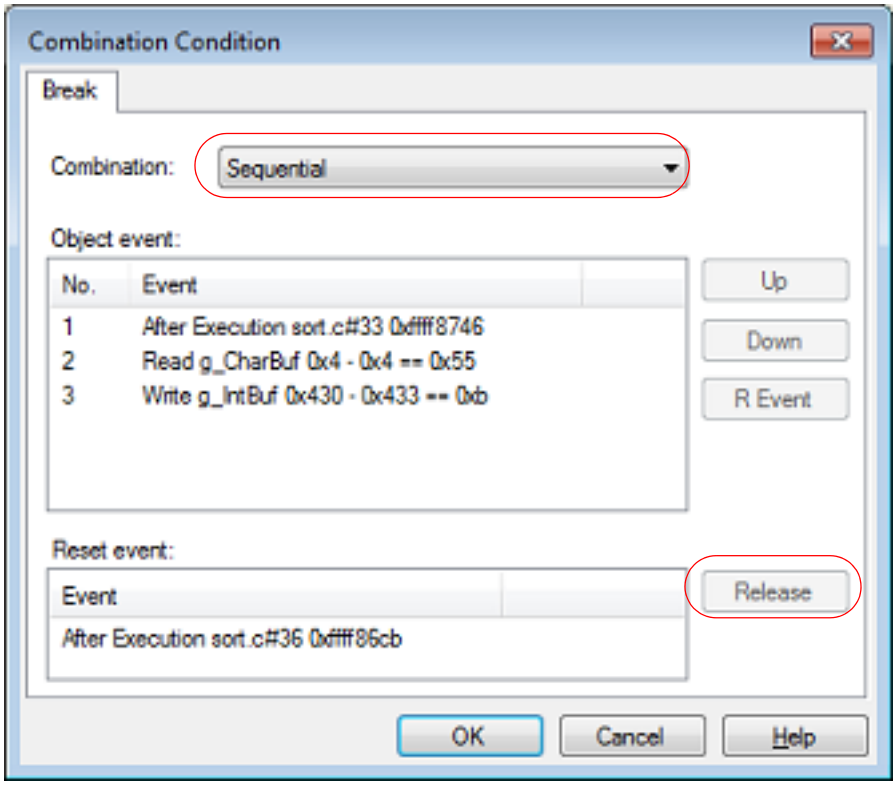


Figure 2.179 Releasing Registration of a Reset Event (Sequential)



2.17.5 Deleting events

To delete any event you've set, select the subject event and then click the toolbar button .

When an event is specified by a combination of multiple events, each event in the detailed information can be deleted independently.

However, you cannot delete the unconditional trace event and Run-Break timer event, which are the built-in events of the debug tool.

Remark 1. For the timer start and end events, and trace start and end events, you can also delete an event by selecting [Delete Event(s)] from the context menu displayed by right-clicking the event mark that is displayed on the Editor panel or Disassemble panel.

Remark 2. 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.

Remark 3. 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.

2.17.7.1 Allowable number of valid events
2.17.7.2 Types of events that can be set or removed during execution
2.17.7.3 Other precautions

2.17.7.1 Allowable 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 2.21 Allowable Number of Valid Events [RX600, RX700 Series]

Type of event	Debug tool	
	E1(Serial)/E1(JTAG) E20(Serial)/E20(JTAG) EZ Emulator	Simulator
Software break	256[Note9]	0
Hardware break (execution-related: before execution)	8[Note1]	1024[Note8]
Break event (execution-related: after execution)	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 Note3]	256
Timer result (timer start, timer end)	8+4[Note1 Note2 Note6]	1[Note7]
Action (Printf event)	256[Note9]	1024[Note8]
Action (Interrupt event)	0	256

Table 2.22 Allowable Number of Valid Events [RX200 Series, RX140 Group]

Type of event	Debug tool	
	E1(Serial) E20(Serial) EZ Emulator	Simulator
Software break	256[Note9]	0
Hardware break (execution-related: before execution)	4[Note4]	1024[Note8]
Break event (execution-related: after execution)	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]	1[Note7]
Action (Printf event)	256[Note9]	1024[Note8]
Action (Interrupt event)	0	256

Table 2.23 Allowable Number of Valid Events [RX100 Series other than RX140 Group]

Type of event	Debug tool	
	E1(Serial) E20(Serial) EZ Emulator	Simulator
Software break	256[Note9]	0
Hardware break (execution-related: before execution)	4[Note4]	1024[Note8]
Break event (execution-related: after execution)	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)	0	1[Note7]
Action (Printf event)	256[Note9]	1024[Note8]
Action (Interrupt event)	0	256

Note 1. There are 8 execution-related events and 4 access-related events, which are shared by multiple functions.

Note 2. For an event combination, up to 8 events can be set.

Note 3. **[E20(JTAG)]**
The trace function and the real-time 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.

Note 4. There are 4 execution-related events and 2 access-related events, which are shared by multiple functions.

Note 5. 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, RX700 Series]**
  7 events + reset event (R event)
- **[RX100, RX200 Series]**
  3 events + reset event (R event)

Note 6. If the maximum number of execution-related events have already been used as trace start/trace end events or timer start/timer end 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.

Note 7. For each of timer start and end events, only a single execution-related event or access-related event can be specified.

Note 8. Hardware breaks (execution-related: before execution) and actions (Printf events) are shared by multiple functions.

Note 9. Software breaks and actions (Printf events) are shared by multiple functions.

2.17.7.2 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) EZ Emulator	Simulator
Software break	△ Note1 Note2	—
Hardware break (execution-related: before execution)	△ Note2	○
Break event (execution-related: after execution)	△ 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 execution
— : Not supported
× : Impossible

Note 1. Internal RAM area only

Note 2. Events cannot be set while the microcontroller is in a power-down mode.

2.17.7.3 Other precautions

- No events can be set to local variables.

- No events occur during step execution (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 2.180 [Hook Transaction Settings] Category

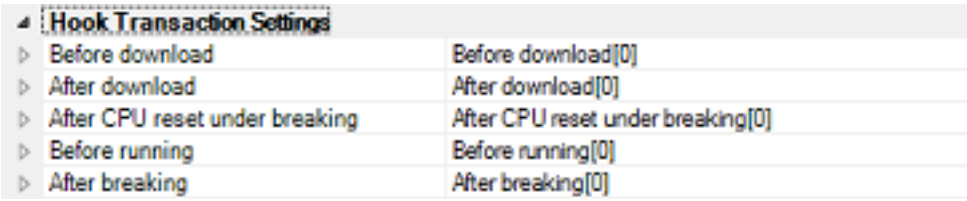


Table 2.25 Properties of the [Hook Transaction Settings] Category

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 2.181 Text Edit Dialog Box Opened

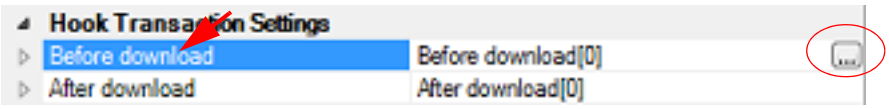
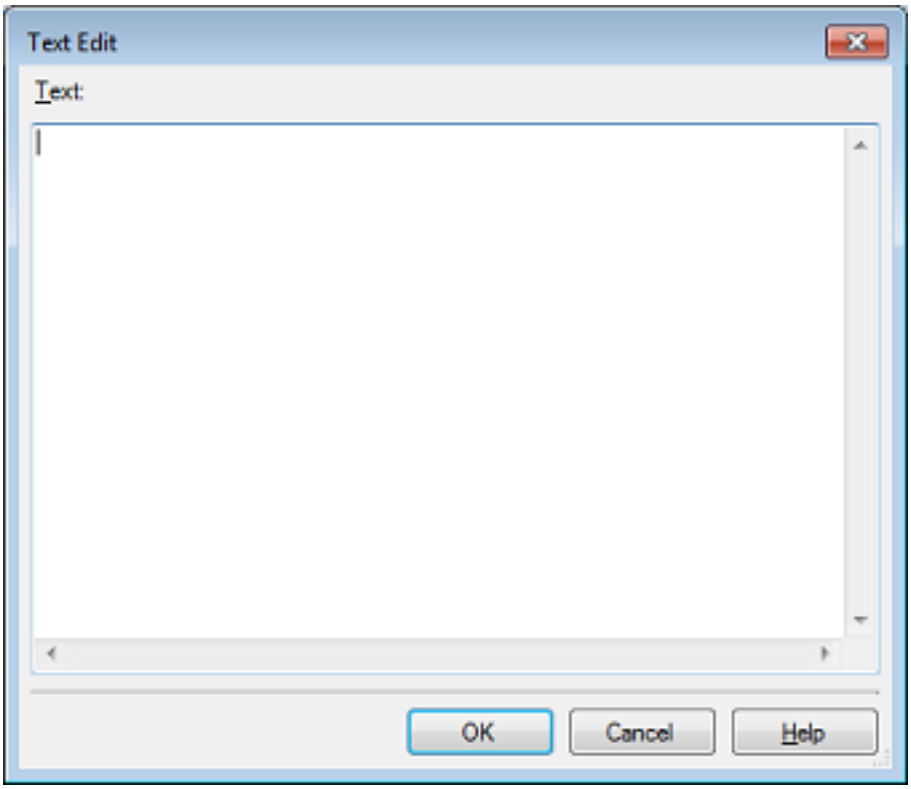


Figure 2.182 Configuring Hook Transaction (Text Edit Dialog Box)



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
```

Remark 1. When specifying the transaction, add "#" at the top of the line to comment it out.

Remark 2. The space can be substituted by a tab character.

Caution The following commands can be used in a Python script to be executed as a hook process of the debugger.

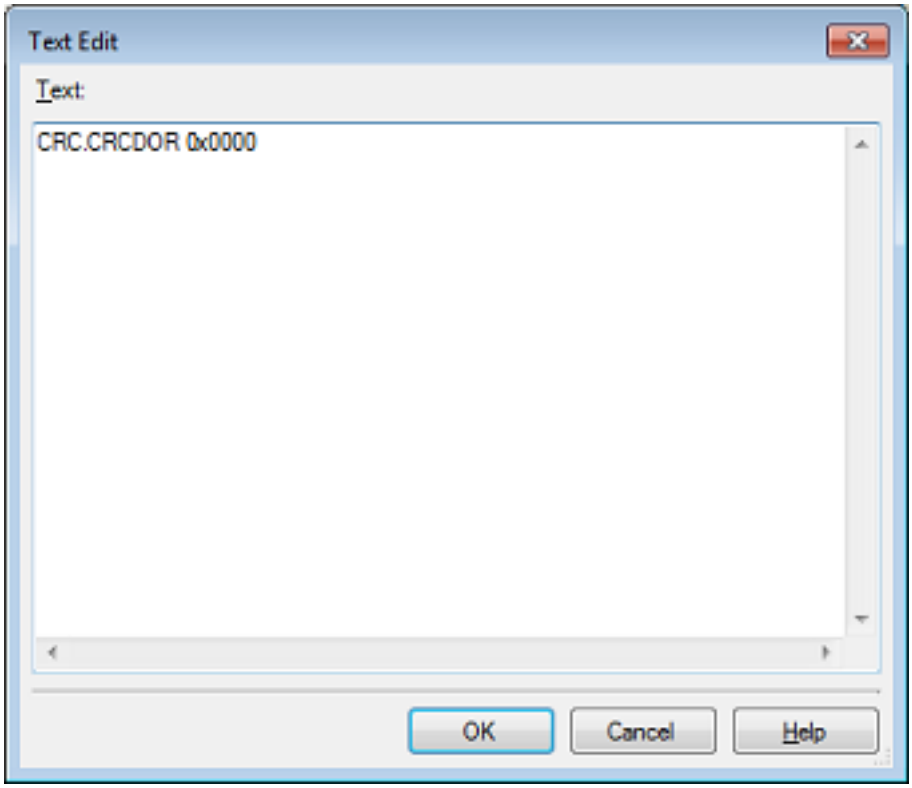
debugger.Register.GetValue
debugger.Register.SetValue
debugger.Memory.GetValue
debugger.Memory.SetValue

If you wish to execute other Python commands, use the Hook command of the Python console.

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 2.183 Example of Hook Transaction Setting



2.19 Measuring Current Consumption [E2]

Measurement of current consumption refers to measuring the amount of current consumption by a whole system to be debugged.

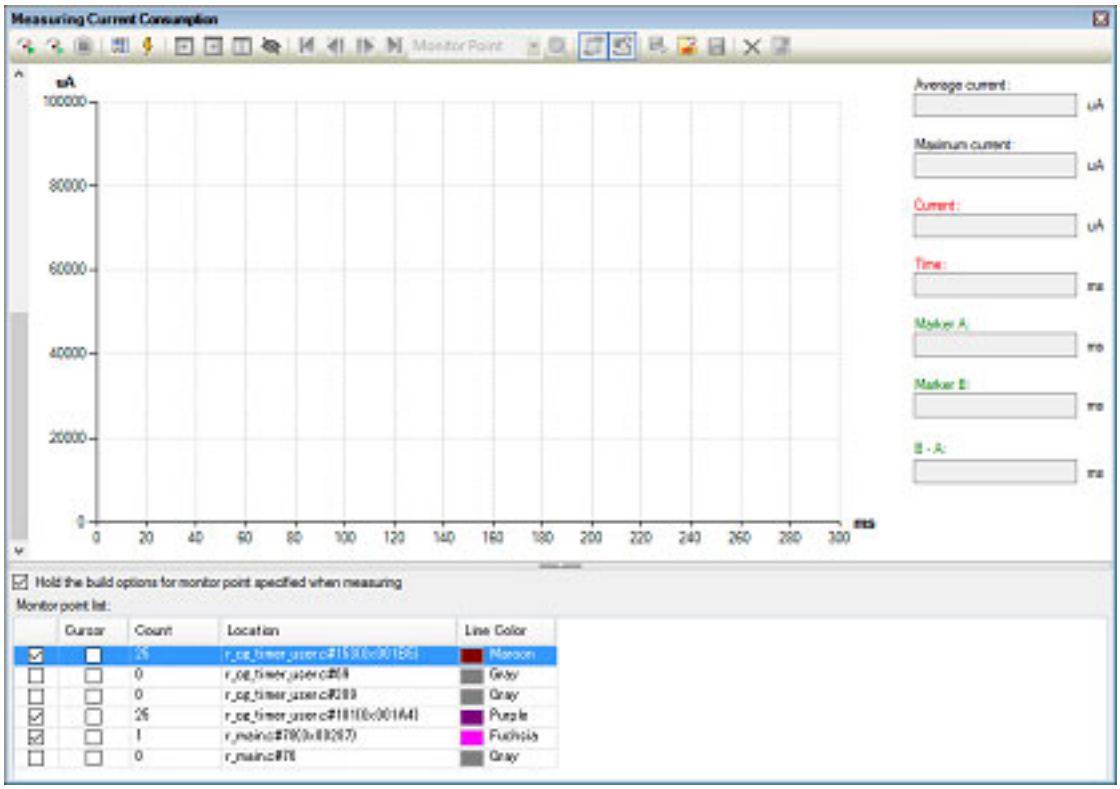
Caution Measuring current consumption is only available for the RX100 and RX200 Series.

[How to use]

In using the E2 emulator to measure current consumption, the current values in the target system as a whole are measured.
For measurement of current consumption, the facility for supplying power to the target by the E2 emulator and the E2 expansion interface are used.
Accordingly, to use this facility, be sure to specify [Yes] for the [Power target from the emulator (MAX 200mA)] property and [Use supplied power from the emulator] for the [Interface for supplying the power] property on the [Connect Settings] tab in the Property panel before connecting the debug tool.

(1) Open the Measuring Current Consumption panel
Select [RX E2] as the debug tool, open the Solution List panel, and click on the [GO] button for measuring current consumption. The Measuring Current Consumption panel [E2] will open.

Figure 2.184 Measuring Current Consumption Panel [E2]



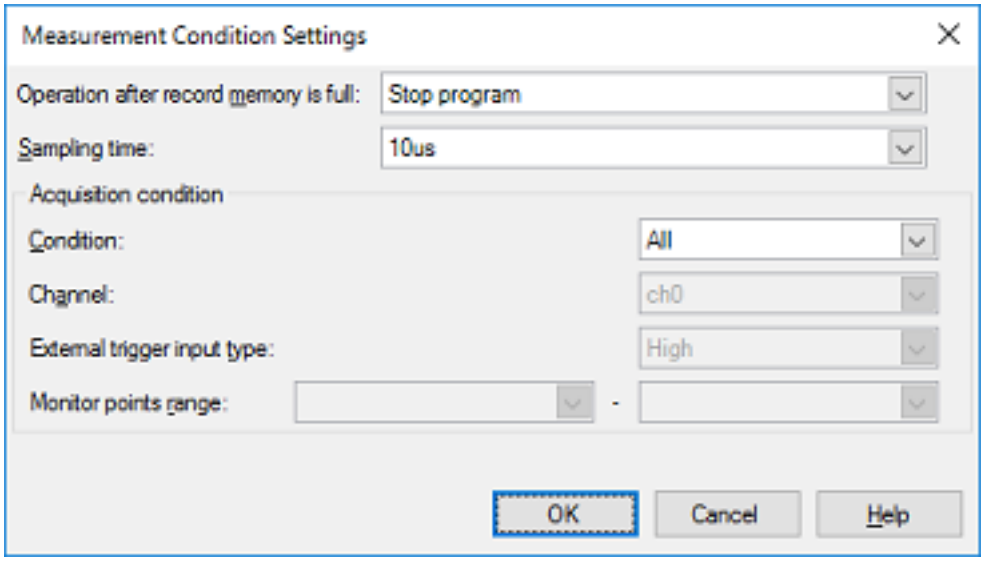
(2) Set the conditions (measurement conditions, trigger conditions, and monitoring points)
In measuring current consumption, you can acquire the pattern of transitions for the current value to be detected by setting conditions in advance. The main settings that can be made in advance are measurement conditions, trigger conditions, and monitoring points.

Remark Conditions are set as follows by default.
Operation after record memory is full: Overwrite to the record memory and continue execution
Sampling time: 10us
Acquisition condition: All
No external trigger is used.
No monitoring points are set.

If you only want to measure the current as a trial, start the procedure from "(3) Start measurement".

(a) Set the measurement conditions
Set conditions regarding the recording of the measured current values in the Current Consumption Measurement Condition Setting dialog box [E2] which is displayed by clicking on on the toolbar in the Measuring Current Consumption panel [E2].

Figure 2.185 Current Consumption Measurement Condition Setting Dialog Box



<1> Select the operation after recording memory becomes full
Select any operation from among the following after the memory used to record the current values in the E2 emulator becomes full.

- Overwrite to the record memory and continue execution
- Stop recording (continue to execute the program)
- Stop program

<2> Select the sampling time
Select the sampling time for use in measurement from among the following.
1us, 2us, 5us, 10us, 20us, 50us, 100us, 200us, 500us, 1ms

<3> Select the acquisition condition
An acquisition condition for recording the current values in the memory of the E2 emulator can be set so that recording only starts when the specific condition has been satisfied.
The condition can be selected as one of the following.

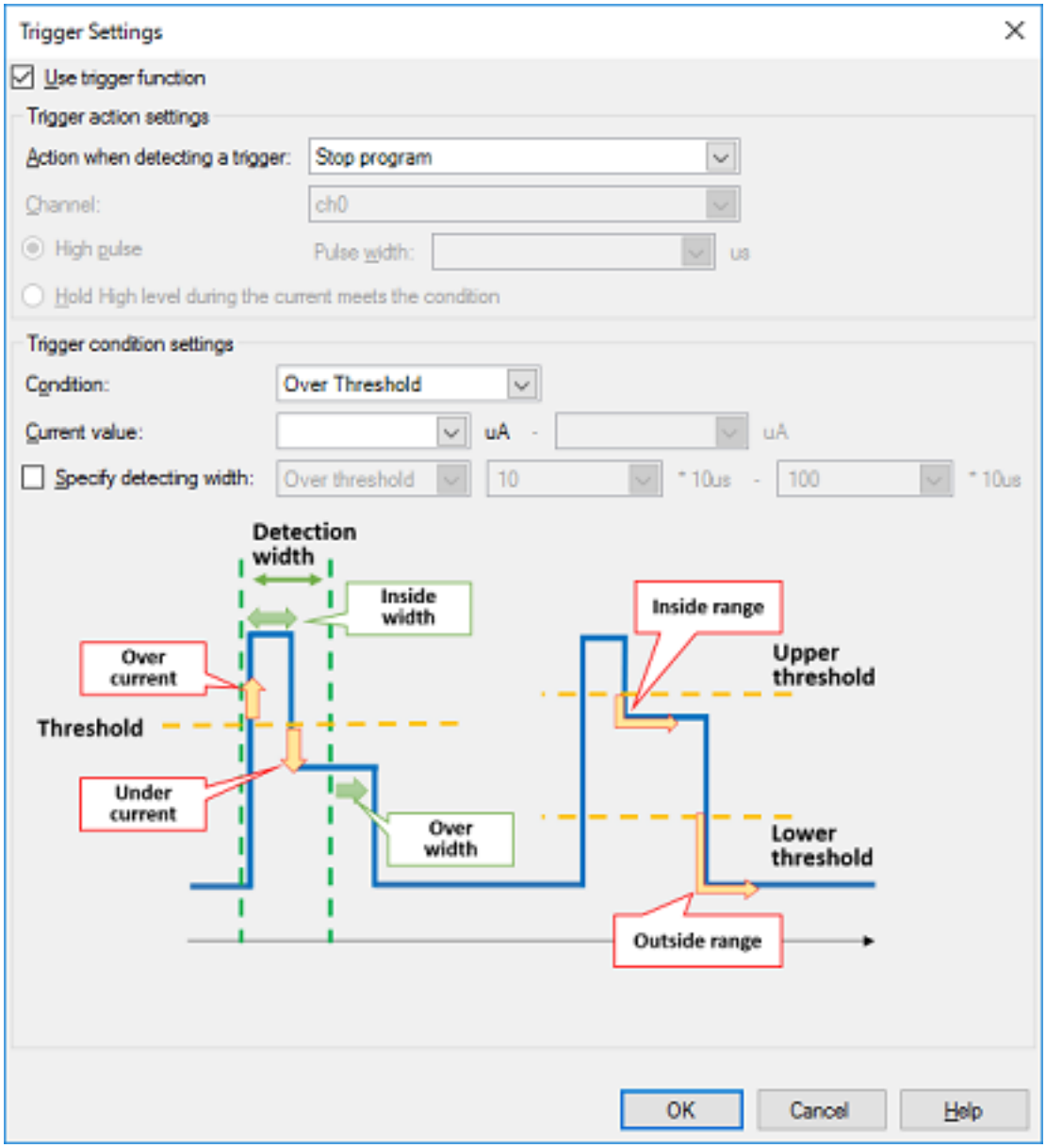
- All
  Record all current values that have been sampled after starting measurement without specifying specific conditions.
- During detecting an external trigger input
  This item selects recording in memory in response to the external trigger input facility of the E2 expansion interface only while the specified signal is being detected.
  When this is specified, the channel for input of the external trigger to be detected and the level of the signal to be detected must be specified.
  In addition, the external trigger input must have been correctly set before the debug tool is connected.

  Remark For details on setting the external trigger input facility of the E2 expansion interface, refer to the Application Note on the Current Consumption Tuning Solution (E2 Emulator, CS+).
- During monitor points
  Specify the monitoring point to serve as the start condition in the left entry field and that to serve as the end condition in the right entry field against [Monitor points range].
  The current values will be recorded from detection of the monitoring point specified as the start condition until detection of the monitoring point specified as the end condition.
  For details on the monitoring points, refer to "(c) Set monitoring points".

(b) Set the trigger condition
The E2 emulator is able to stop the program or output an external trigger signal in response to the value for current being measured satisfying a specific condition.

Set the trigger condition in the Current Consumption Measurement Trigger Condition Setting dialog box [E2], which is displayed by clicking on on the toolbar in the Measuring Current Consumption panel [E2].

Figure 2.186 Current Consumption Measurement Trigger Condition Setting Dialog Box



<1> Specify whether to use the trigger facility
Select the checkbox for [Use trigger function] to use the trigger facility.
This checkbox is de-selected by default.

<2> Set the trigger action
Select the operation when a trigger is detected as either of the following.

- Stop program
  Stop execution of the program.
- Output external trigger signal
  Output an external trigger through the E2 expansion interface.

Also make the following settings if an external trigger is to be output.

- Select a channel
  Specify the channel for output of the external trigger through the E2 expansion interface.
- Select the type of signal to be output

Select whether a high pulse is to be output in response to the condition being satisfied or output of the high level is to continue as long as the condition is being satisfied.
When a high pulse is to be output, also specify the pulse width.

Remark For details on the external trigger output facility, refer to the Application Note on the Current Consumption Tuning Solution (E2 Emulator, CS+).

<3> Set the trigger condition
Select the trigger condition for the current value as [Over Threshold], [Under Threshold], [In Range], or [Out of Range].
Specify the current value or range of current values to be detected as the trigger.

Caution For a range of current values, the value entered in the left field for entry must be lower than that in the right field.

As well as the condition for the current value, a sub-condition (condition for the detection width) regarding the time over which the condition is satisfied can be set. Specifying a width for detection can eliminate the detection of noise that temporarily satisfies the condition.
The condition for the detection width is selectable as [Over threshold] (detecting the current value when the condition continues to be satisfied for more than a specified time) or [In range] (detecting the current value when the condition is satisfied over a time within the specified range).

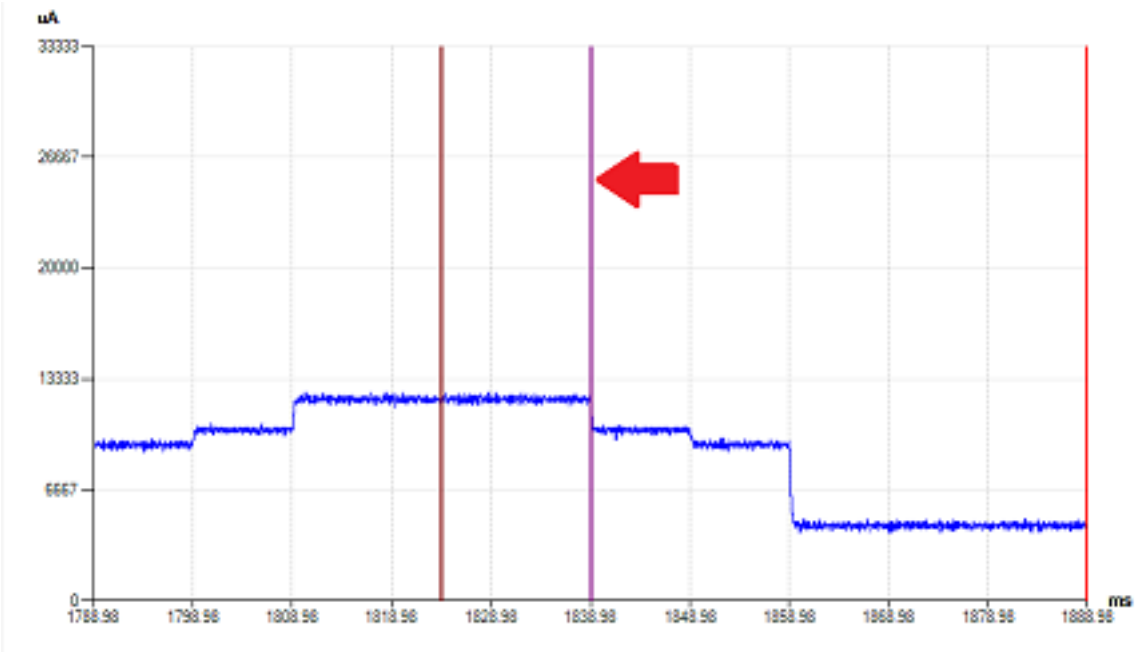
Specify a multiple of the sampling time as the detection width.
Note that the actual time for detection also depends on the sampling time.

(c) Set monitoring points
A monitoring point is a facility for recording a timestamp when a specified line of source code is run.
When monitoring points are set, transitions of the execution time and the current values at the monitoring points can be confirmed on a graph for measuring current consumption.

Caution Do not use the debug console while you are using monitoring points.

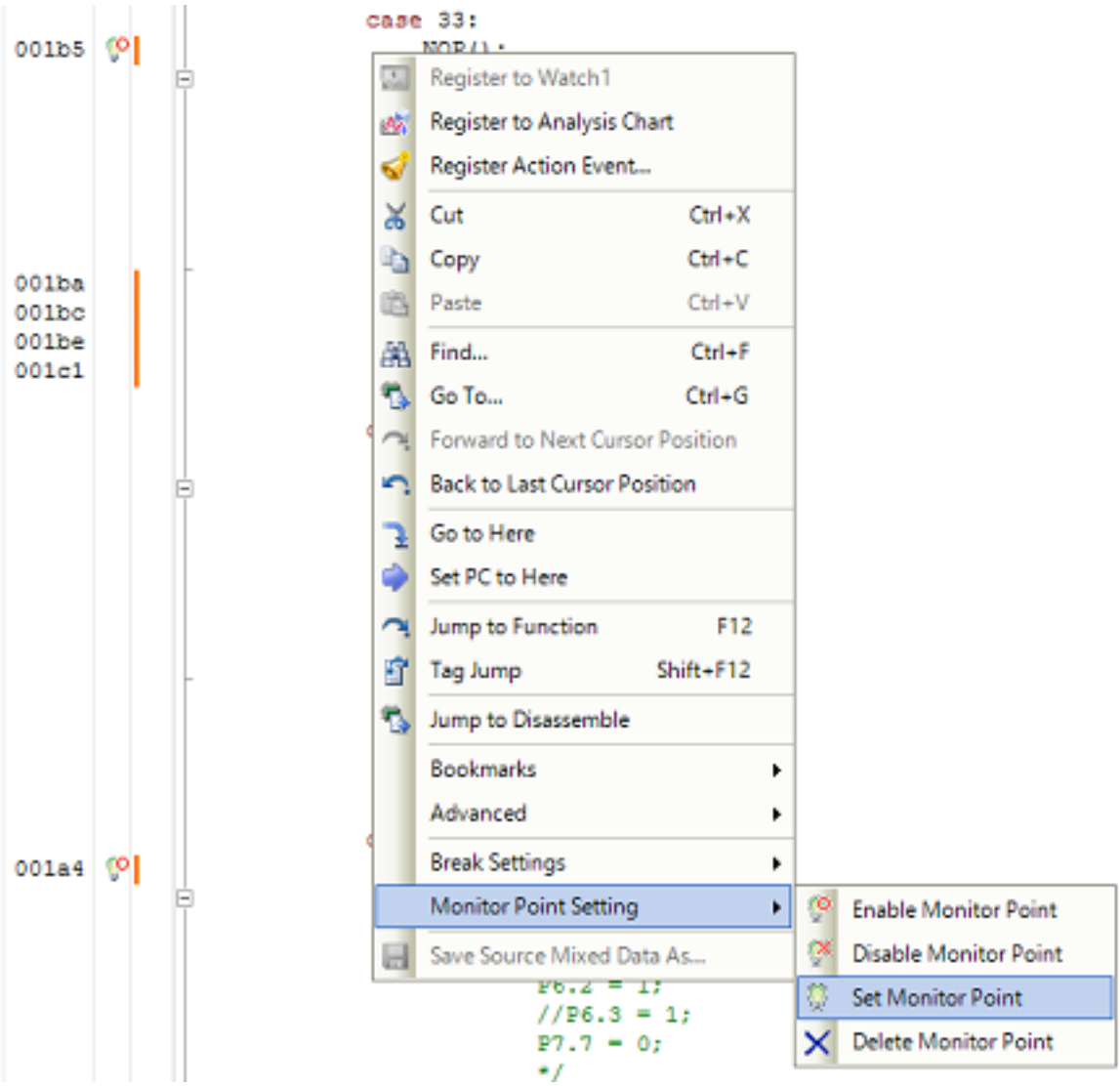
The monitoring points are indicated by colored vertical lines on the graph.

Figure 2.187 Monitoring Point



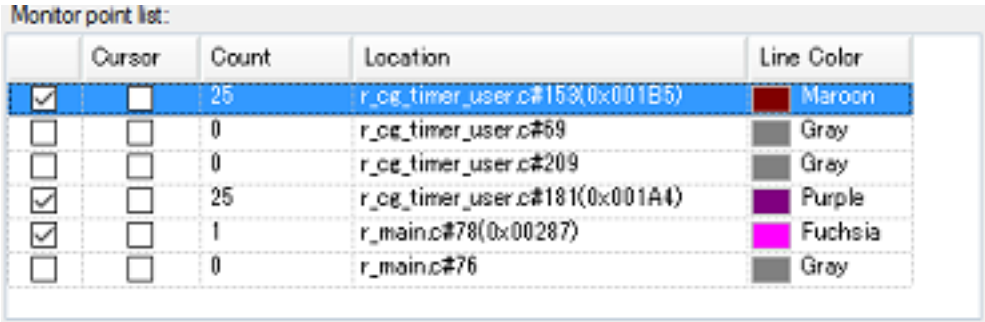
Monitoring points are set from [Monitor Point Setting] in the context menu after placing the cursor on the line of the target source code that was opened in the Editor panel.

Figure 2.188 Context Menu in the Editor Panel



A list of the monitoring points that are currently set is displayed under the Measuring Current Consumption panel [E2].
The monitoring points that have been set are enabled or disabled by the leftmost checkbox in this list or from the context menu of the Editor panel.

Figure 2.189 List of Monitoring Points



Caution 1. In the following cases, the monitoring points cannot be set when measurement starts, so that they are forcibly disabled.

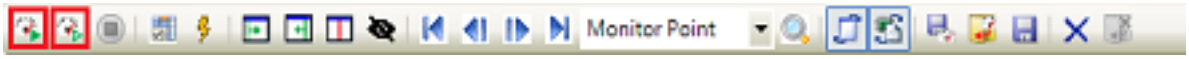
- When monitoring points are set at instructions other than NOP instructions or the debug tool cannot detect the NOP instruction even if it is set as a monitoring point

- In the [System] category on the [Debug Tool Settings] tab in the Property panel, when [Yes] is selected either for the following properties:
  - [Execute the specified routine immediately before execution of the user program]
  - [Execute the specified routine immediately after the user program stops]
  - [Debug the program re-writing the on-chip PROGRAM ROM]

Caution 2. Monitoring points must be set at the addresses of NOP instructions.
When the compiler in use for the project is V2.08.00 or later of CC-RX and when the source file is registered in the active project, the NOP instructions can be automatically inserted at the addresses where the monitoring points are set before measurement and measurement can be started. However, when the [CPU Reset after download] property in the [Download] category on the [Download File Settings] tab in the Property panel is set to [Yes], note that the monitoring points are forcibly reset by rebuilding and downloading.

(3) Start measurement
Start measuring current consumption by the target system by using the facilities of the E2 emulator.
Click on or on the toolbar in the Measuring Current Consumption panel [E2].

Figure 2.190 Toolbar in the Measuring Current Consumption panel [E2]



Caution Current consumption is not measured by clicking on in the Main window.

(a) Clicking on
The project is rebuilt and downloaded before measurement starts.
However, if the device does not support monitoring points or the compiler used in the active project is a version of CC-RX earlier than V2.08.00, rebuilding and downloading do not proceed.
When [Hold the build options for monitor point specified when measuring] in the Measuring Current Consumption panel [E2] is selected, the options which are specified for rebuilding and downloading before measurement are reflected in the [Parameters of NOP instruction insertion for measuring current consumption] property of the build tool.

(b) Clicking on
The project is not rebuilt and downloaded before measurement starts.
Select this button when you want to measure current consumption without resets due to rebuilding and downloading.
Note that this button is not displayed if the device does not support monitoring points or the compiler used in the active project is a version of CC-RX earlier than V2.08.00.

(4) End measurement
Measurement ends at the same time as execution of the program is stopped.
Execution of the program can be stopped by setting a breakpoint or a trigger condition, or by clicking on .

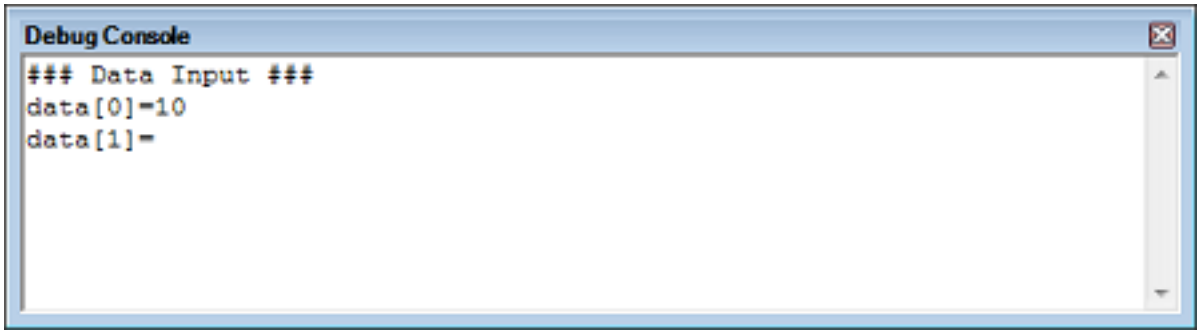
(5) Display a graph
At the same time as measurement ends, a graph of current values showing the results of measurement is displayed in the graph display area of the Measuring Current Consumption panel [E2].
Check that the current values that are shown are as intended.

In the panel, dragging a cursor (the red vertical line) shows the time and the current value at the cursor location in the information display area.
When the search facility is used, since the cursor is moved to the location where the search condition was satisfied, the time and the current value at that location can be confirmed.
Markers A and B (green vertical dashed lines) can also be dragged to confirm the average current and maximum current between the markers.
When a monitoring point is the target of the search, the location where processing to be confirmed was executed can be identified immediately.

2.20 Using the Debug Console

This section describes how to use the debug console for standard I/O.
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 2.191 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 I/O 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.

CS+ 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 I/O.
  - Initialize I/O 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] [EZ Emulator]
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.
When the debug console is used, real-time performance of the user program is affected due to transmission or reception of data.

Remark 1. For details about the low-level interface routines, see "CC-RX Compiler User's Manual".

Remark 2. **[Simulator]**
As a method for executing standard I/O, an emulator-like method and a simulator-inherent method are supported. In a simulator-inherent method, file I/O, not just standard I/O, can also be used.
Each supported method changes from one to another as you specify in the [Select stream I/O mode] property in the [Stream I/O] [Simulator] category on the Property panel's [Debug Tool Settings] tab.
For details about the simulator's I/O functions, see "B. I/O 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 CS+, load sample project "RX610_Tutorial_DebugConsole" for the E1/E20.

Remark For details about the "Start panel," see the separate edition, "CS+ Integrated Development Environment User's Manual: Project Operation."

(2) Editing the low-level interface routine file (assembly language part) [Simulator]
To execute stream I/O in simulator mode, you need to edit the content of a file (lowlvl.src) for low-level interface routines (assembly language part) to make it usable for a simulator.
Replace the contents of "lowlvl.src" with the following sample code for the simulator.

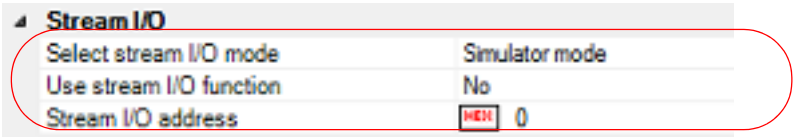
```
          .GLB     _charput
          .GLB     _charget
SIM_IO    .EQU 0h
          .SECTION    P,CODE
;----------------------------------------------------------------------
;  _charput:
;----------------------------------------------------------------------
_charput:
          MOV.L       #IO_BUF,R2
          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 2.192 [Stream I/O] Category



(a) [Select stream I/O mode]
To use files for the same low-level interface routines (assembly language part) as the emulator, specify [Emulator mode]. (By default, [Simulator mode] is specified.)

(b) [Use stream I/O function]
This property is displayed only when [Simulator mode] is specified in the [Select stream I/O mode] property.
If you intend to perform standard I/O or file I/O, select [Yes] (default selection: [No]).

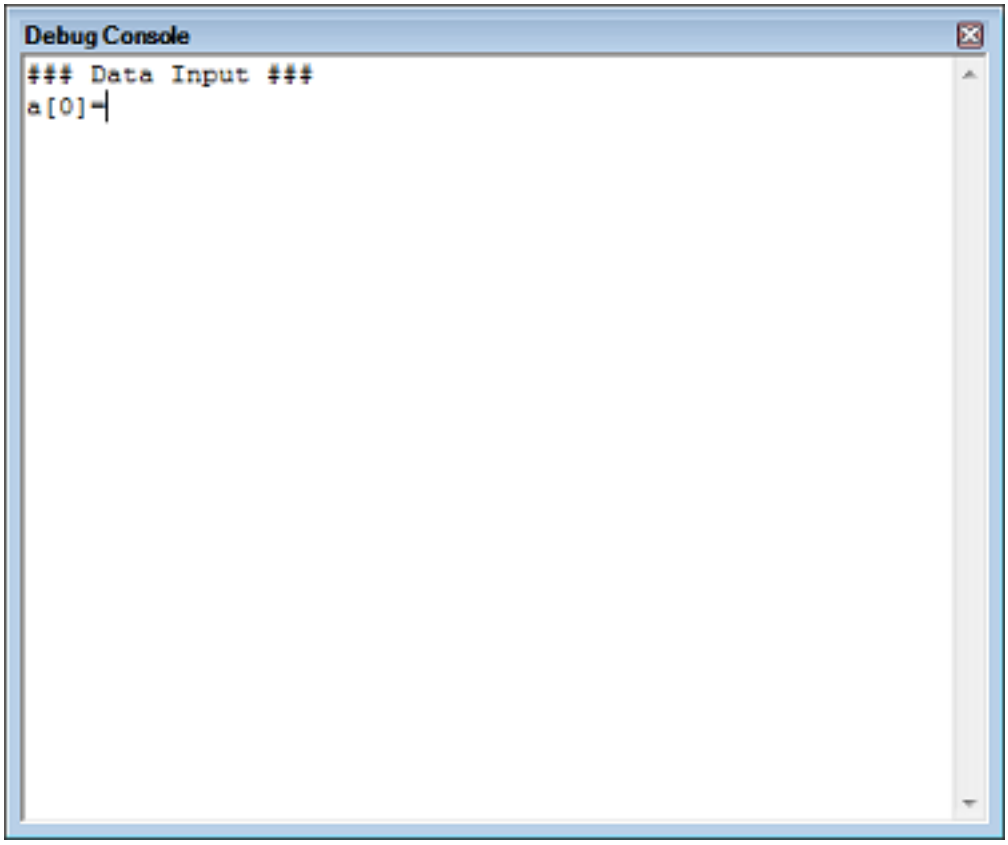
(c) [Stream I/O address]
This property is displayed only when [Simulator mode] is specified in the [Select stream I/O mode] property.
If you have replaced the low-level interface routines (assembly-language part) with the sample code for the simulator, enter "0" (default).

(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.

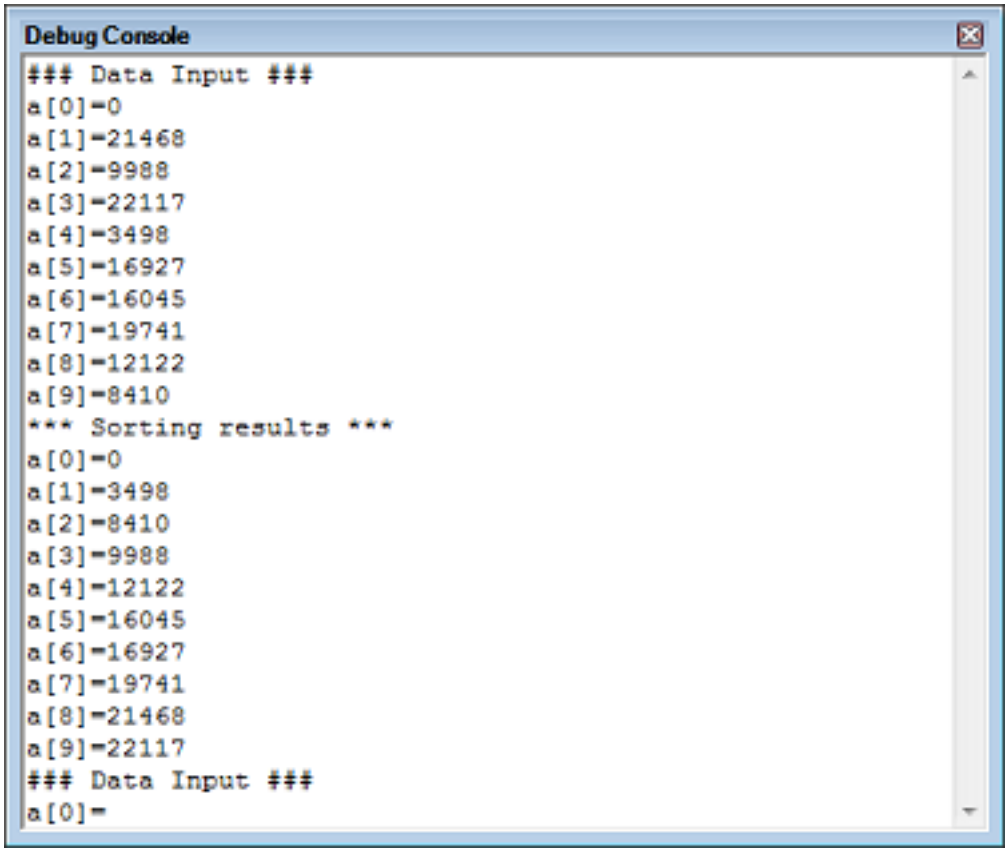
(7) Executing the program
Execute the program (see "2.9 Execute Programs").

(8) Performing data I/O on Debug Console panel
By executing the program, perform data I/O 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.)

2.21 About Input Value

This section describes consideration to take when inputting values in each panel and dialog box.

2.21.1 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	Starts with 0b and continues with the numbers from 0 to 1. (Case insensitive for alphabets)
Octal number	Starts with 0 and continues with the numbers from 0 to 7.
Decimal number	Starts without 0 and continues with the numbers from 0 to 9.
Hexadecimal number	Starts 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
C/C++ variable name Note 1 No 2	Address of a C/C++ variable
Expression [*Expression*Note 3]	Address of an array
Expression.Member nameNote 4	Address of a structure/union/class member
Expression->Member nameNote 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
CPU register name	Value of a CPU register
I/O register name	Address of an I/O register
Label nameNote 5, EQU symbol nameNote 5, [immediate value]	Address of a label, a value of an EQU symbol, and an immediate address
Integer constant	Address

Note 1. It represents C89, C99 and C++ variables.

Note 2. If the register is assigned the value of a C variable, an error results.

Note 3. The expression that is input as an index to an array is parsed as a watch-expression.

Note 4. To specify a member variable of base class, specify scope before a member name.
(Example: variable.BaseClass::member)

Note 5. If a label name or EQU symbol name contains a "$," enclose the name in braces "{ }."
(Example: {$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 sign
!*Expression*	Logical negation
~*Expression*	Bit inversion
Expression * *ExpressionNote*	Multiplication
Expression / *Expression*[Note]	Division
Expression % *Expression*[Note]	Remainder calculation
Expression + *Expression*[Note]	Addition
Expression - *Expression*[Note]	Subtraction
Expression & *Expression*[Note]	Bit-wise logical AND
Expression ^ *Expression*[Note]	Bit-wise logical exclusive OR
Expression | *Expression*[Note]	Bit-wise logical (inclusive) OR

Note Variables and functions can be combined by an operator only with variables, functions and integer constants. (Example: C variable name + I/O register 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
C/C++ variable name[Note 1]	Value of a C/C++ variable
Expression[*Expression*]	Element values of an array
Expression.Member name[Note 2]	Member values of a structure/union/class
Expression->Member name[Note 2]	Member values of the structure/union/class pointed to by a pointer
Watch-expression.*Cast expression	Value of a pointer to member variable
Watch-expression->*Cast expression	Value of a pointer to member variable
**Expression*	Value of pointer variable
&*Expression*	Location address
CPU register name	Value of a CPU register
I/O register name	Value of an I/O register
Label name[Note 3], EQU symbol name[Note 3], [immediate value]	Value of a label, value of an EQU symbol, value of an immediate address
Integer constant	Constant value of an integer
Floating constant	Constant value of a floating point
Character constant	Constant value of a character

Note 1. It represents C89, C99 and C++ variables.

Note 2. To specify a member variable of base class, specify scope before a member name. (Example: variable.BaseClass::member)

Note 3. If a label name or EQU symbol name contains a "$," enclose the name in braces "{ }." (Example: {$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*	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]	Bit-wise logical AND
Expression ^ *Expression*[Note]	Bit-wise logical exclusive OR
Expression | *Expression*[Note]	Bit-wise logical (inclusive) OR

Note Variables and functions can be combined by an operator only with variables, functions and integer constants. (Example: C variable name + I/O register name)

Caution 1. When two or more variables are defined with the same name in unnamed name spaces of different source files, the Watch panel only shows the information on the first variable to be found.

Caution 2. After the member-variable pointer "mp1" defined in the program below is registered with the Watch panel and Local Variables panel, the type of the pointer is indicated as "int *", not "int Foo::*".

```
class Foo {
    int m1;
};
int Foo::*mp1 = &Foo::m1;
```

Caution 3. When a char-type one-dimensional array is assigned to multiple locations in registers or memory as shown below, no character string will be displayed in the value column of the Watch panel or Local Variables panel even after the array "array" has been registered with the panel.
char array[5] = "ABCD";

Caution 4. When the selection for [Scope] in the Local Variables panel is not [Current], the values of variables assigned to registers are not displayed correctly. Editing these values is also not possible.

Caution 5. When a defined variable satisfies both of the conditions given below, the Watch panel and Local Variables panel indicate the location of the entire variable rather than the location of its member variables.

[Conditions]

<1> The variable is assigned to two or more addresses or registers (i.e. two or more addresses or registers are displayed in the address column).

<2> A structure-, class-, array-, or union-type member is defined in the variable.

[Example]

```
struct Mem {
    long m_base;
};
struct Sample {
    long m_a;
    struct Mem m_b; <- Condition <2>
};

main () {
    struct Sample obj;
}
```

Display in the Watch and Local Variables panels:

```
"obj"                          -    { R1:REG, R2:REG }      (struct Sample)
    L m_a            0x00000000    { R1:REG }              (long)
    L m_b                          -    { R1:REG, R2:REG }      (struct Base)
        L m_base     0x00000000    { R2:REG }              (long)
```

Caution 6. When a variable is cast to another type in the Watch panel, casting of the variable is C-style.
For this reason, the result of casting a class using virtual inheritance to its base class is not the same as the result of the cast within the program.

```
class AAA [
    int m_aaa;
} objA;
class BBB : public AAA {  //BBB inherits AAA.
    int m_bbb;
} objB;
class CCC {  //CCC does not inherit AAA.
    int m_ccc;
} objC

class AAA* pa = objA;
class BBB* pb = objB;
class CCC* pc = objC;

"(AAA*)pa"  Usable
"(BBB*)pb"  Usable
"(AAA*)pb"  Usable
"(CCC*)pc"  Usable
"(AAA*)pc"  Not usable because pc is considered to point to the top address
of AAA.
Image of the cast in a program: (AAA*)((void*)pc)
```

Caution 7. In case of a variable with the size beyond 0x10000, Value, Style and Address indicate "?" in the Watch panel.

2.21.2 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.

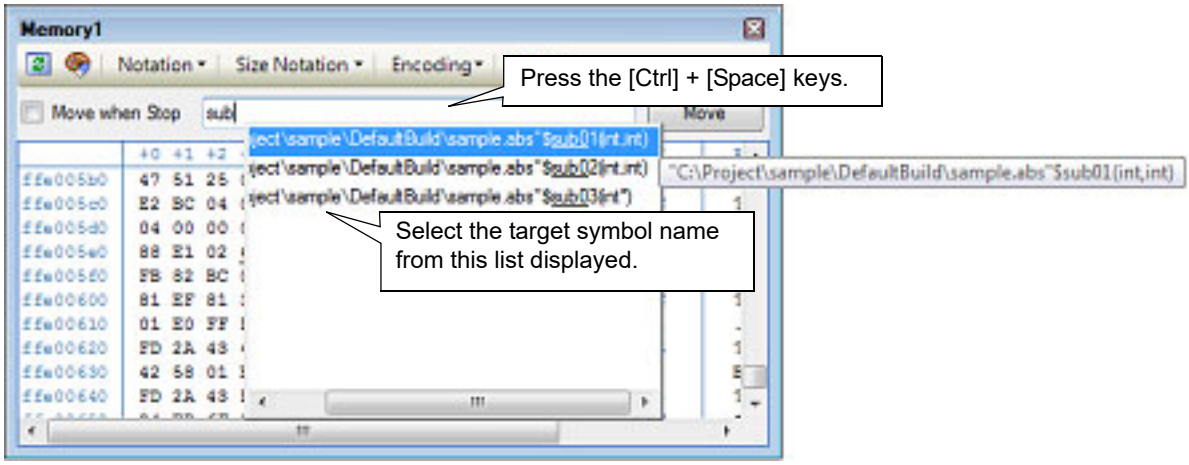
At this time, if a key other than the [Space]/[Enter] key is pressed or the focus moves to outside the panel/dialog box currently being operated, then the list of symbol names will disappear (the symbol name completion will not be performed).

Caution 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.

Caution 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 2.193 Symbol Name Completion Function



2.21.3 Icons for invalid input

In some of the dialogs provided by CS+, 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.

2.22 Differences between the Microcontroller and the Emulator

For differences between the microcontroller and the emulator, see "E1/E20/E2 Emulator, E2 Emulator Lite Additional Document for User's Manual (Notes on Connection of RX)".

2.23 Other Notes on Usage [E1] [E20] [EZ Emulator]

For other notes on usage, see "E1/E20/E2 Emulator, E2 Emulator Lite Additional Document for User's Manual (Notes on Connection of RX)".

A. WINDOW REFERENCE

This appendix describes in detail the windows, panels and dialog boxes used when debugging with CS+.

A.1 Description

The table below lists the windows, panels and dialog boxes associated with debugging.

Table A.1 List of Window, Panel and Dialog Box

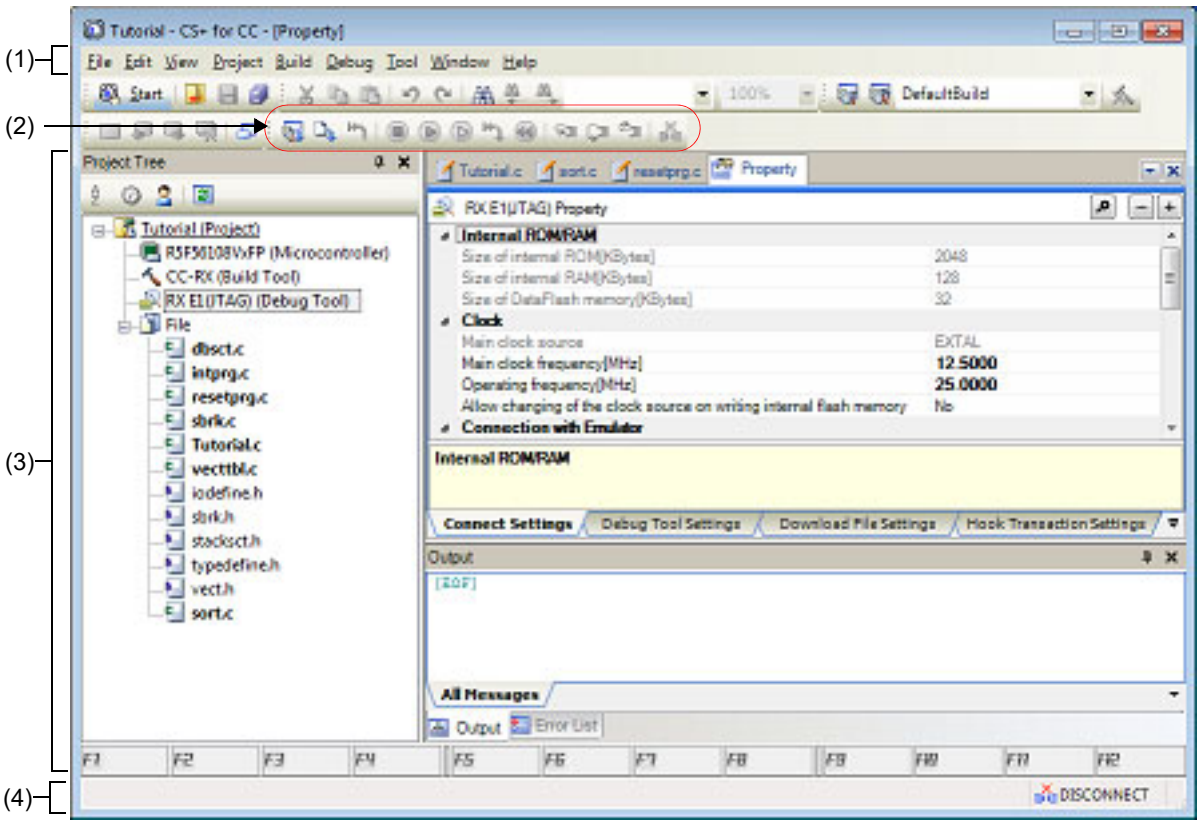
Window, panel and dialog-box names	Functional description
Main window	Controls program execution and opens each panel.
Project Tree panel	Selects the debug tool to be used.
Property panel	Displays detailed information on a debug tool selected on the Project Tree panel and changes settings.
Memory panel	Displays memory values and changes values.
Disassemble panel	Displays disassembled result of memory values and performs line assembly and instruction-level debugging.
CPU Register panel	Displays contents of CPU registers (program and system registers) and changes values.
IOR panel	Displays contents of I/O registers and changes values.
Local Variables panel	Displays contents of local variables and changes values.
Watch panel	Displays contents of registered watch-expressions and changes values.
Call Stack panel	Displays information on call stack used in function calls.
Trace panel	Displays trace data acquired from the debug tool.
Events panel	Displays detailed information on set events, chooses to enable or disable events, and deletes events.
Output panel	Displays messages output by the build tool, debug tool or each plug-in, or displays result of global search performed in Find and Replace dialog box.
Debug Console panel	Performs standard I/O.
Memory Mapping dialog box	Sets memory mapping.
Download Files dialog box	Selects a file to be downloaded and sets download condition.
Action Events dialog box	Sets action events.
Column Number Settings dialog box	Sets the number of view columns of memory values on the Memory panel.
Address Offset Settings dialog box	Sets an offset value for the address display on the Memory panel.
Memory Initialize dialog box	Initializes memory.
Memory Search dialog box	Searches memory.
Print Address Range Settings dialog box	Sets a print range on the Disassemble panel.
Trace Search dialog box	Searches for trace data.
Combination Condition dialog box [E1] [E20] [EZ Emulator]	Displays the conditions for event combination and modifies the settings.

Window, panel and dialog-box names	Functional description
Detailed Settings of Timer Measurement dialog box [E1] [E20] [EZ Emulator]	Displays the detailed information on timer events and modifies the settings.
Detailed Settings of Execution Events dialog box	Displays the detailed information on execution-related events and modifies the settings.
Detailed Settings of Access Events dialog box	Displays the detailed information on access-related 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 a scroll range on the Memory panel/Disassemble panel.
Go to the Location dialog box	Moves the caret to a specified position.
Data Save dialog box	Saves displayed content of each panel and upload data.
Functions and Variables Access Table panel	Displays the functions that access variables in the form of an orthogonal table.
Measuring Current Consumption panel [E2]	This panel is central to the functionality of the solution for measuring current consumption.
Current Consumption Measurement Condition Setting dialog box [E2]	This dialog box is used to set measurement conditions in the Measuring Current Consumption panel [E2].
Current Consumption Measurement Trigger Condition Setting dialog box [E2]	This dialog box is used to set trigger conditions in the Measuring Current Consumption panel [E2].
Current Consumption Measurement Search dialog box [E2]	This dialog box is used to specify and search the detailed conditions in the Measuring Current Consumption panel [E2].
Save Current Values dialog box [E2]	This dialog box is used to save the values for current in the Measuring Current Consumption panel [E2].

Main window

This is the first window that opens when you launch CS+.
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]

- Select Windows [Start] >> [Programs] >> [Renesas Electronics CS+] >> [CS+ for CC (RL78,RX,RH850)].

Remark In Windows 8.1, select [CS+ for CC (RL78,RX,RH850)] on the start screen.
In Windows 10, select Windows [Start] menu >> [All apps] >> [Renesas Electronics CS+] >> [CS+ for CC (RL78,RX,RH850)].

[Description of each area]

(1) Menu bar
The debug-related menu items are described 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 CS+ 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 CS+ is disconnected from the debug tool.
Call Stack	Opens the Call Stack panel. However, this item is disabled when CS+ is disconnected from the debug tool.
Memory	Shows the following cascaded menu to open the Memory panel. However, this item is disabled when CS+ 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 CS+ is disconnected from the debug tool.
CPU Register	Opens the CPU Register panel. However, this item is disabled when CS+ is disconnected from the debug tool.
Trace	Opens the Trace panel. However, this item is disabled when CS+ is disconnected from the debug tool.
Disassemble	Shows the following cascaded menu to open the Disassemble panel. However, this item is disabled when CS+ 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 CS+ is disconnected from the debug tool.
Debug Console	Opens the Debug Console panel. However, this item is disabled when CS+ 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 CS+ 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 "2.6.2.4 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.

(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 CS+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. This item is disabled during program execution/build (not including rapid build) 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 CS+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. This item is disabled during program execution/build (not including rapid build) execution. When the rebuild has failed, download will not be executed.
Rebuild & Download	Rebuilds a project and executes a download to the debug tool currently selected in the active project after the rebuild is complete. If CS+ is disconnected from the debug tool at this time, it is automatically connected to the debug tool before a download is executed. This item is disabled during program execution/build (not including rapid build) execution. When the rebuild has failed, download will not be executed.
Connect to Debug Tool	Connects to the debug tool currently selected in the active project. This item is disabled while connected to the debug tool, during build (not including rapid build) execution or if the version of compiler being used is not supported by CS+.
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 CS+ using hot plug-in [E1(JTAG)] [E20(JTAG)]"). 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 (not including rapid build) is under execution, or when disconnected from the debug tool.
Disconnect from Debug Tool	Terminates communication with the currently connected debug tool. However, this item is disabled when build (not including rapid build) is under execution or when already disconnected from the debug tool.
Using Debug Tool	The following cascade menus are displayed to select the debug tool to use. Note that the debug tools displayed in this menu differ depending on the microcontroller selected in the project.
RX E2	Uses E2 as the debug tool.
RX E2 Lite	Uses E2 Lite as the debug tool.
RX E1(Serial)	Uses E1 in Serial communication mode as the debug tool.
RX E1(JTAG)	Uses E1 in JTAG communication mode as the debug tool.
RX E20(Serial)	Uses E20 in Serial communication mode as the debug tool.
RX E20(JTAG)	Uses E20 in JTAG communication mode as the debug tool.
RX EZ Emulator	Uses EZ Emulator as the debug tool.
RX Simulator	Uses Simulator as 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 (not including rapid build) 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 (not including rapid build) is under execution, or when disconnected from the debug tool.
Step In	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, the program stops at the beginning of a called function. However, this item is disabled during program execution, when build (not including rapid build) 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 execution will continue until the same nest is formed as when a jump to subroutine instruction has been 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 (not including rapid build) 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 (not including rapid build) 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 (not including rapid build) 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 (not including rapid build) is under execution or when disconnected from the debug tool.
Rewind debug tool state	Rewinds the debug tool to the last state that was automatically saved. Note that the data to be rewound is limited to memory and register values that can be read or written. To use this debugging function, it has to be set so in the Option dialog box.
Save debug tool state	The following menus are relevant to saving and restoring of the state of the debug tool. Note that the data to be saved is limited to memory and register values that can be read or written.
Restore debug tool state *n*	Restores the state of the debug tool from the *n*-th data file.
Save debug tool state *n*	Saves the current state of the debug tool in a file as the *n*-th data.

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).

Remark 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.

Remark 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 CS+ 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 CS+ 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/build (not including rapid build) execution. This is the same as selecting [Download] from the [Debug] menu.
	Resets the CPU (program is not executed). However, this item is disabled when build (not including rapid build) is under execution or when disconnected from the debug tool. This is the same as selecting [CPU Reset] from the [Debug] menu.
	Forcibly halts the currently executed program. However, this item is disabled when the program is already halted or disconnected from the debug tool. This is the same as selecting [Stop] from the [Debug] menu.
	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 (not including rapid build) is under execution, or when disconnected from the debug tool. This is the same as selecting [Go] from the [Debug] menu.
	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 (not including rapid build) is under execution, or when disconnected from the debug tool. This is the same as selecting [Ignore break and go] from the [Debug] menu.
	Resets the CPU once and then starts running the program from the reset address. However, this item is disabled when build (not including rapid build) is under execution or when disconnected from the debug tool. This is the same as selecting [Restart] from the [Debug] menu.
	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). For a function call, the program stops at the beginning of the called function. However, this item is disabled during program execution, when build (not including rapid build) is under execution, or when disconnected from the debug tool. This is the same as selecting [Step In] from the [Debug] menu.
	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). 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 execution will continue until the same nest is formed as when a jump to subroutine instruction has been executed). For other than a jump to subroutine instruction, the same operation as the button is clicked is performed. However, this item is disabled during program execution, when build (not including rapid build) is under execution, or when disconnected from the debug tool. This is the same as selecting [Step Over] from the [Debug] menu.

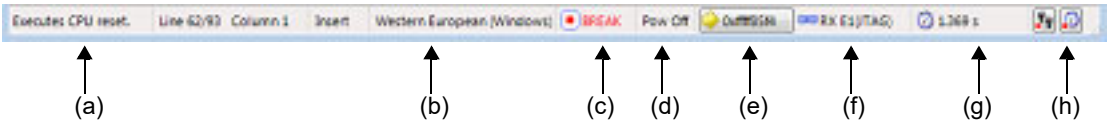
	Runs the program until control returns from the currently executed function (until control returns to the calling function)[Note] (Return Out execution). However, this item is disabled during program execution, For other than a jump to subroutine instruction, the same operation as the when build (not including rapid build) is under execution, or when disconnected from the debug tool. This is the same as selecting [Return Out] from the [Debug] menu.
	Terminates communication with the currently connected debug tool. However, this item is disabled when build (not including rapid build) is under execution or when already disconnected from the debug tool. This is the same as selecting [Disconnect from Debug Tool] from the [Debug] menu.

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 Status Bar



(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 CS+ 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 CS+ 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

Remark Nothing is displayed here when the CPU is in status other than those listed above.

(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.
Note that the content displayed during program execution changes according to settings in the [Register] category on the Property panel's [Debug Tool Settings] tab, as follows:

Program state	State of [PC display during the execution] property	Displayed content
While halted	--	Current PC position
During execution	Yes	Current PC position at update intervals set in [Display update interval for PC[ms]] property Note 1
	No	“Running“

Placing the cursor over this area will pop up the following information.

- Current PC: 0x *current PC value* (*source name # number of lines* Note 2)

However, this status is hidden when CS+ is disconnected from the debug tool.

Note 1. [RX100 Series]
PC values are hidden because these microcontrollers do not support display of the PC value in the status bar during program execution.

Note 2. If this information cannot be obtained, it is substituted for by "Symbol name + Offset value".

(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	*Debug tool name*
Currently disconnected	DISCONNECT

(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 CS+ is disconnected from the debug tool.

Status	Displayed content
No measurements mode	Not measured
Measurement in progress	Measuring
When overflowed	OVERFLOW

Remark When the EZ Emulator is in use, the status shown in the status bar is always “Not measured” because the EZ Emulator does not support the Run-Break timer function.

(h) Debug tool status
Displays the current status of each function of the debug tool using the following icons.

However, this status is hidden when CS+ is disconnected from the debug tool.

Function	In operation	Not in operation (Enable)	Disable
Trace			
Timer			-
Coverage [Simulator][[E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]			

Caution 1. **[E1] [E20] [EZ Emulator]**
The trace function does not go to a "Disable" state even when you click the subject icon.

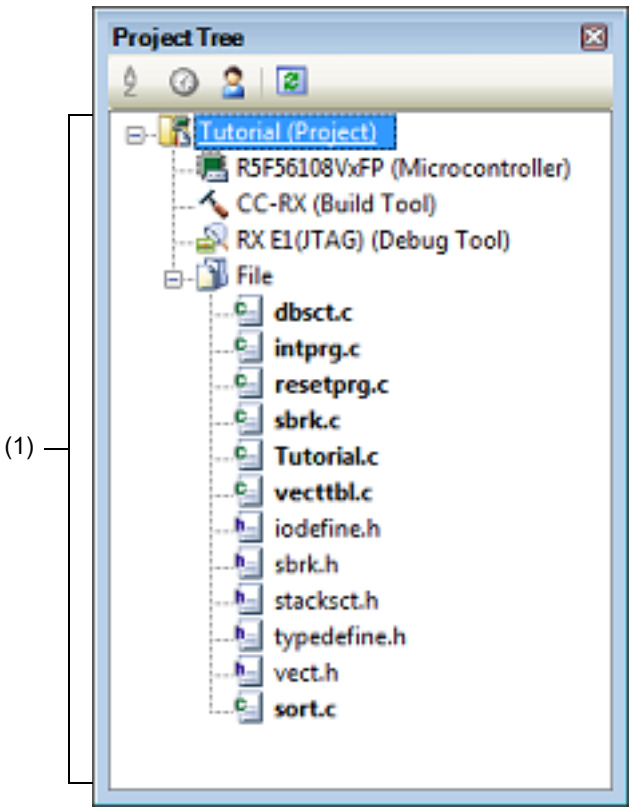
Caution 2. **[E20(JTAG)]**
When you click the icon of the trace function, the contents set in the [Usage of trace function] property in the [Trace] category on the Property panel's [Debug Tool Settings] tab is toggled between Trace and Real-time RAM monitor.

Caution 3. **[Simulator]**
When the program is halted, you can click the subject icons of the trace and coverage functions to change their "Enable" or "Disable" state.
Also, the set contents of the following properties on Property panel's [Debug Tool Settings] tab toggle between Yes and No.
- [Use trace function] property in the [Trace] category
- [Use coverage function] property in the [Coverage] [Simulator] category

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]
- [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
Microcontroller type Debug tool name (debug tool)	- *Microcontroller type* Shows the type name of a selected microcontroller (RX). - *Debug tool name* Shows the debug tool name used in the project (E2, E2 Lite, E1(Serial), E1(JTAG), E20(Serial), E20(JTAG), EZ Emulator, 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.)

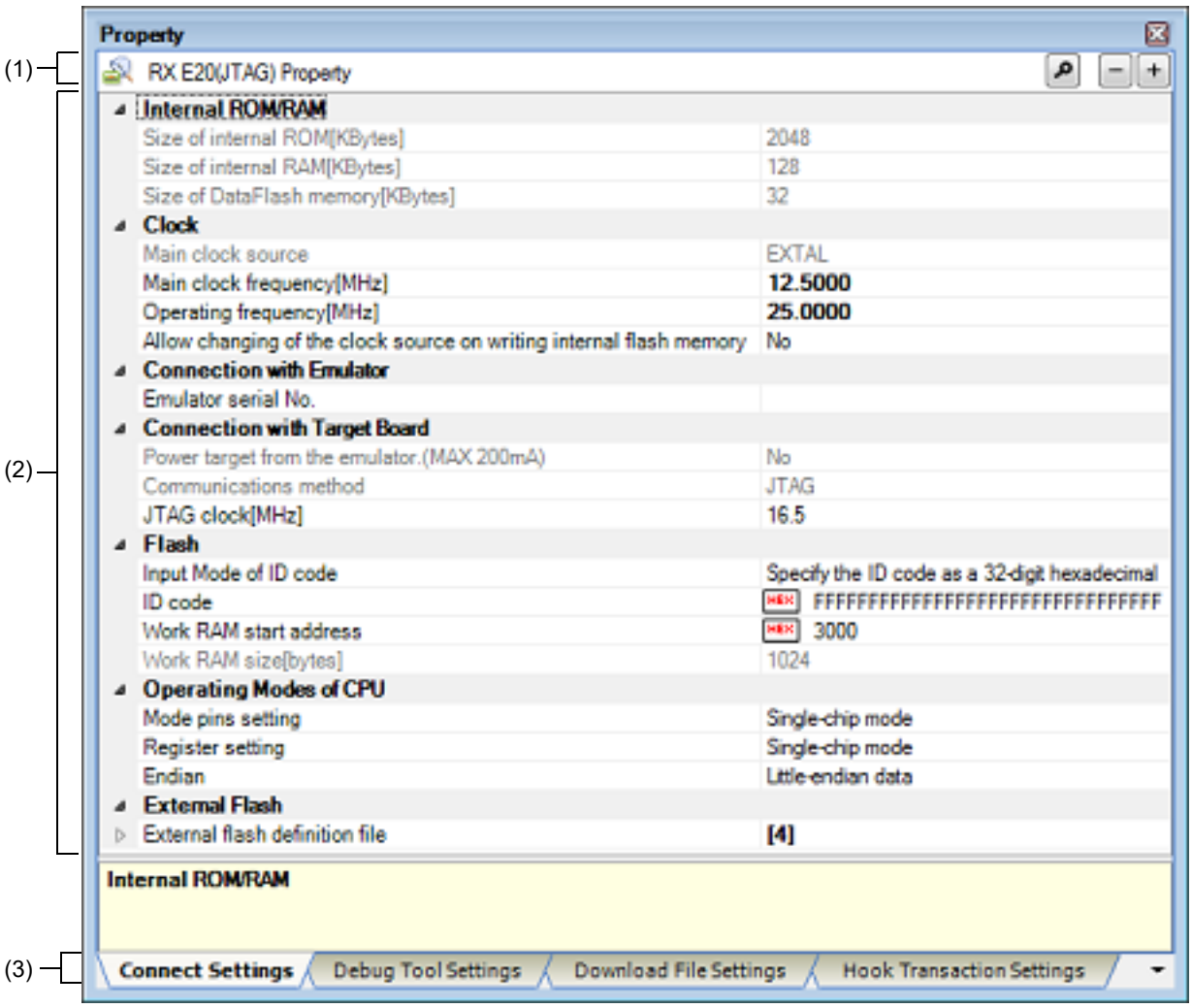
[Context menu]

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 E2	Uses E2.
Microcontroller type E2 Lite	Uses E2 Lite.
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 EZ Emulator	Uses EZ Emulator.
Microcontroller type Simulator	Uses the simulator.
Property	Displays the properties of the selected debug tool on the Property panel.

Property panel

This panel displays per-category detailed information about a debug tool selected on the Project Tree panel and changes the setting made in it.

Figure A.4 Property Panel (Example When E20(JTAG) is Selected)



This section describes the following.

- [How to open]
- [Description of each area]
- [[Edit] menu (Property Panel-Only Items)]
- [Context menu]

[How to open]

- On the Project Tree panel, select the [*Microcontroller type Debug tool name* (Debug Tool)] node that you use, and then choose [Property] from the [View] menu or select it on the context menu.
- On the Project Tree panel, double-click the [*Microcontroller type Debug tool name* (Debug Tool)] node that you use.

Remark If this panel is already open, detailed information on a selected debug tool is displayed on it when you select the [*Microcontroller type Debug tool name* (Debug Tool)] node that you use on the Project Tree panel.

[Description of each area]

(1) Subject name area
This area displays the name of a debug tool that is selected on the Project Tree panel.

(2) Detailed information display/change area
This area displays detailed information on a debug tool selected on the Project Tree panel in list form classified by category, allowing you to change settings directly.
The ⊟ mark denotes that all property items included in the category concerned are expanded. The ⊞ mark denotes that the property items in the category are collapsed. The display can be switched between expanded and collapsed views by clicking either of these marks or by double-clicking a category name.
Note that the HEX mark displayed in each property item setting column denotes that the text box marked with it is used exclusively for hexadecimal input.
For details about the displayed contents of categories and those of property items included therein, and on how to set them, see the section in which the relevant tab is described.

(3) Tab selection area
Each time you select a tab in this area, the category for which detailed information is displayed changes from one to another.
The following tabs are accommodated on this panel. (For details about the displayed contents on each tab, and on how to set them, see the section in which the relevant tab is described.)

- [Connect Settings] tab
- [Debug Tool Settings] tab
- [Download File Settings] tab
- [Hook Transaction Settings] tab

[[Edit] menu (Property Panel-Only Items)]

Undo	Cancels the editing work on a property value that you've done immediately before.
Cut	Cuts a selected character string and moves it to the clipboard, if you use this menu while editing a property value.
Copy	Copies the character string of a selected property value to the clipboard.
Paste	Inserts the content of the clipboard into place, if you use this menu while editing a property value.
Delete	Deletes a selected character string, if you use this menu while editing a property value.
Select All	Selects the whole character string representing the value of a selected property, if you use this menu while editing a property value.

[Context menu]

[Except when editing a character string]

Reset to Default	Reverts the set value of a selected property item to the default.
Reset All to Default	Reverts all set values on the currently selected tab to the default.

[When editing a character string]

Undo	Cancels the editing work on a property value that you've done immediately before.
Cut	Cuts a selected character string and moves it to the clipboard, if you use this menu while editing a property value.
Copy	Copies the character string of a selected property value to the clipboard.

Paste	Inserts the content of the clipboard into place, if you use this menu while editing a property value.
Delete	Deletes a selected character string, if you use this menu while editing a property value.
Select All	Selects the whole character string representing the value of a selected property, if you use this menu while editing a property value.

[Connect Settings] tab

The [Connect Settings] tab displays detailed information for each category shown below and changes settings made on it.

(1) [Internal ROM/RAM]
(2) [Endian] [Simulator]
(3) [Clock]
(4) [Connection with Emulator] [E1] [E20]
(5) [Connection with Target Board] [E1] [E20] [EZ Emulator]
(6) [E2 Expansion Interface] [E2]
(7) [Flash] [E1] [E20] [EZ Emulator]
(8) [Operating Modes of CPU] [E1] [E20] [EZ Emulator]
(9) [External Flash] [E1] [E20]
(10) [Peripheral Function Simulation] [Simulator]

[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	- For products with internal ROM *Internal ROM size of a selected microcontroller* - For ROM-less products *0*
	How to change	Not changeable
Size of internal RAM[Kbytes]	Displays the internal RAM size of a selected microcontroller.	
	Default	*Internal RAM size of a selected microcontroller*
	How to change	Not changeable
Size of DataFlash memory[Kbytes] [E1] [E20] [EZ Emulator]	Displays the size of the data flash memory area of a selected microcontroller.	
	Default	*Data flash memory size of a selected microcontroller*
	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	*Little-endian data*
	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: Little-endian data or Big-endian data

(3) [Clock]
This category displays detailed information on clock and changes clock settings.

Main clock source **[E1] [E20] [EZ Emulator]**	- 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] [EZ Emulator]**	Specify EXTAL frequency in MHz units. Note that this property is enabled only when [EXTAL] is specified in the [Main clock source [E1] [E20] [EZ Emulator]] 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)	
Operating frequency[MHz] **[E1][E20] [EZ Emulator]**	Specify the Operating frequency (ICLK) in MHz units.		
	Default	*Blank*	
	How to change	By entering directly from the keyboard	
	Specifiable value	0.0001 to 999.999 (in MHz)	
Allow to change the clock source on writing internal flash memory [E1] [E20] [EZ Emulator]	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	Manipulation of the main clock source is allowed.
		No	Manipulation of the main clock source is not allowed.
System clock (ICLK) frequency[MHz] [Simulator]	Specify the CPU's operating clock frequency.		
	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	An integer in the range 1 to 1,000. (in MHz)	

(4) [Connection with Emulator] [E1] [E20]

Emulator serial No.	Select the serial No. of the emulator to be connected.[Note]	
	Default	*Blank*
	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.
If an attempt is made to connect the emulator when this category is blank, the serial number of the emulator that is found first by search will be automatically selected and connection is made. The serial number of the emulator that is automatically selected in such a case will not be saved in the project information.

(5) [Connection with Target Board] [E1] [E20] [EZ Emulator]
This category displays detailed information on a state of connection with the target board and changes connection settings.

User Vcc[V] [EZ Emulator]	Specify the value of the microcontroller's operating voltage that is actually supplied to the user board.		
	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 9.9999 (in V)	
Power target from the emulator (MAX 200mA) **[E1] [E20]**	Specify whether power is supplied from E1 to the target system. 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	Power is supplied.
		No	Power is not supplied.
Interface for supplying the power [E2]	Select the interface for supplying the power to the target board from the emulator.		
	Default	USER I/F	
	Modifying	Select from the drop-down list.	
	Available values	USER I/F	Uses the user interface.
		E2 expansion I/F	Uses the E2 expansion interface.

Supply voltage [E1]	Specify the voltage supplied from E1 to the target board. Note that this property is displayed only when you selected [Yes] in the [Power target from the emulator (MAX 200mA) [E1] [E20]] property.	
	Default	*3.3V*
	How to change	- [E1][E2 Lite] By selecting from the drop-down list. However, changeable only when disconnected from the debug tool. - [E2] Select from the drop-down list or directly enter from the keyboard.
	Specifiable value	- [E1] One of the following as selected from the drop-down list: Note *3.3V, 5.0V* - [E2 Lite] *3.3V* - [E2] Directly enter the numbers ranged below 1.8 to 5.0 (unit: V)
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)] [EZ Emulator] *FINE* - [E2 Lite][E2] *JTAG*
	How to change	- [E1] [E20] [EZ Emulator] Not changeable - [E2 Lite][E2] Not changeable or by selecting from the drop-down list Whether this property can be changed differs with the type of the selected microcontroller and the communication method. See "2.2.1 Confirm the connection to a host machine" for the relationship between types of the microcontroller and the communication method.
	Specifiable value	- [E2 Lite][E2] *JTAG, FINE*

JTAG clock [MHz] [E1(JTAG)] [E20(JTAG)]	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	- [E1(JTAG)] [E20(JTAG)] *16.5* - [E2 Lite] *6.00* - [E2] *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: - [E1(JTAG)] [E20(JTAG)] *16.5, 12.38, 6.188, 3.094, 1.547* - [E2 Lite] *6.00, 3.00, 1.50* - [E2] *16.5, 12.5, 6.25, 3.125, 1.562*
FINE baud rate[bps] [E1(Serial)] [E20(Serial)] [EZ Emulator]	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	- [E1(Serial)] [E20(Serial)][E2] *2000000* - [EZ Emulator] *1000000* - [E2 Lite] *1500000*
	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: - [E1(Serial)] [E20(Serial)][E2] *2000000, 750000, 500000, 250000* - [EZ Emulator] *1000000, 500000, 250000* - [E2 Lite] *1500000, 750000, 500000, 250000*

Note The specifiable voltage value differs with each selected microcontroller.

(6) [E2 Expansion Interface] [E2]
The detailed information on the E2 expansion interface is displayed and its configuration can be changed.

Interface for supplying the power [E2]	Select whether to use the E2 expansion interface.		
	Default	No use	
	Modifying	Select from the drop-down list.	
	Available values	No use	Does not use the E2 expansion interface.
		Use the power supplied from the target	Uses the E2 expansion interface by the power supplied from the target.
		Use supplied power from the emulator	Uses the E2 expansion interface by the supplied power from the emulator.

(7) [Flash] [E1] [E20] [EZ Emulator]
This category displays detailed information on flash memory rewriting and changes rewrite settings.

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	Enter the ID code to release the flash memory from the protected state.		
	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
		To enter the ID code as a 32-digit hexadecimal value, arrange it as a sequence of 32-bit units of data.
		In the example below, the code to be entered is 00112233445566778899aabbccddeeff.
		`const unsigned long __OSISreg[4] = {`
		`0x00112233,`
		`0x44556677,`
		`0x8899aabb,`
		`0xccddeeff`
		`};`
		Byte order of the ID code of the flash programmer can be different from that of CS+. Please confirm the specification about ID code of the flash programmer you're using.
		- For [Specify the ID code as an ASCII code within 16 characters]
		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. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected. 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	*Depends on the selected microcontroller*
	How to change	Not changeable

Note The work RAM area can also be used by the user program because the emulator saves and restores data in this area. Note, however, that the work RAM area is not specifiable as:
the destination or origin of a DMA or DTC transfer, an address where a DTC vector table or transfer information is to be allocated, or the interrupt vector for a DMAC or DTC activation source.

(8) [Operating Modes of CPU] [E1] [E20] [EZ Emulator]
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	*Single-chip mode*	
	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** - [E1] [E20] *Single-chip mode or user boot mode* - [EZ Emulator] *Single-chip mode*	
Allow erasing the USB boot program [E1] [E20]	Specify whether to erase the USB boot program in the user boot area when you are starting up the emulator in the user boot mode by using a microcontroller in which the USB boot program is stored. This property is displayed only when [*User boot mode*] is selected for the [Mode pins setting] property. Note that this is not displayed for an microcontroller in which the USB boot program in the user boot area does not need to be erased when you are starting up the emulator.		
	Default	*No*	
	How to change	By selecting from the drop-down list However, changeable only when disconnected from the debug tool	
	Specifiable value	Yes	Allows erasing the USB boot program in the user boot area
		No	Does not allow erasing the USB boot program in the user boot area

Register setting	Specify the operation mode set by registers.		
	Default	*Single-chip mode*	
	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** - [E1] [E20] *Single-chip mode, on-chip ROM enabled extended mode or on-chip ROM disabled extended mode* - [EZ Emulator] *Single-chip mode*	
Endian	Displays the project's endian. Note 2		
	Default	*Little-endian data*	
	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: *Little-endian data or big-endian data*	
Change the start bank [RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups]	Specify whether to change the start bank before connecting to the debug tool.		
	Default	*No*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Changes the start bank.
		No	Does not change the start bank.
Start bank [RX651-2M, RX65N-2M, RX671, RX72M, RX66N, and RX72N Groups]	Specify the start bank to be changed before connecting to the debug tool.		
	Default	*Bank 0*	
	How to change	By selecting from the drop-down list	
	Specifiable value	- One of the following as selected from the drop-down list: *Bank 0, Bank 1*	

Note 1. The specifiable operation mode differs with each selected microcontroller.

Note 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.

(9) [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. Note The number of external flash definition files that can be registered is displayed as a main property. This property is expanded to display the file name, address range and download conditions as sub-properties. If no external flash definition files are registered, sub-properties other than File are not displayed.		
	Default	- Main item *4* - Sub-items *Blank*	
	How to change	- Main item Not changeable - Sub-items <File> 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. <Start address> Not changeable <End address> Not changeable <Erase external flash ROM before download> By selecting from the drop-down list However, changeable only when disconnected from the debug tool	
	Specifiable value	Yes	The external flash ROM is erased before downloading.
		No	The external flash ROM is not erased before downloading.
	Display content	- Main item Number of external flash definition files - Sub-items By clicking the "+" mark of an external flash definition file, the definition file name, start and end addresses, and whether to erase the external flash ROM before downloading are displayed for each index ([1] to [4]).	

Note Any settings made in this section are not reflected when the microcontroller is operating in single-chip mode.

(10) [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	- Main item *<Number of usable peripheral function simulation modules>* - Sub-items <Peripheral function simulation module name> *Not use*	
	How to change	- 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	- 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

The [Debug Tool Settings] tab displays detailed information for each category shown below and changes settings made on it.

(1) [Memory]
(2) [Access Memory While Running]
(3) [Register]
(4) [E2 Expansion Interface] [E2]
(5) [Break] [E1] [E20] [EZ Emulator]
(6) [System] [E1] [E20] [EZ Emulator]
(7) [Trace]
(8) [Timer] [E1] [E20] [EZ Emulator]
(9) [Coverage] [Simulator]
(10) [Stream I/O] [Simulator]
(11) [Execution Mode] [Simulator]
(12) [Instruction Decode Cache] [Simulator]
(13) [Coverage] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]

[Description of each category]

(1) [Memory]
This category displays detailed information on memory and changes memory settings.
For details about the displayed types of memory, see the section where the Memory Mapping dialog box is described.

Memory mappings	Displays the current state of memory mapping per type[Note 1] of memory area.		
	Default	*[Sum total by microcontroller's inherent type of memory mapped area]*	
	How to change	Specification by Memory Mapping dialog box The Memory Mapping dialog box is opened by clicking the [...] button that is displayed at the right edge in the column when this property is selected. (It is not possible to change mapping on this panel.)	
	Display Contents	Displays the state of memory mapping per type of memory area. Note that when the "+" mark for each memory type is clicked, the next detailed information is displayed. - Memory type - Start address - End address - Access width[bits] **[E1] [E20] [EZ Emulator]** [Note 2] - Endian[Note 2 Note 3]	
Verify on writing to memory [E1] [E20] [EZ Emulator]	Specify whether or not to perform a verify check when memory values are initialized.		
	Default	*Yes*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	A verify check is performed.
		No	A verify check is not performed.

Note 1. This refers to the type of memory-mapped areas registered in a device file.

Note 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.

Note 3. The endian information differs in displayed contents and specifiable values depending on [Memory Type].

- External area **[E1] [E20] [EZ Emulator]**
  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]
This category displays detailed information on memory access during program execution and the realtime display update function (see "2.11.1.4 Displaying and changing memory contents during program execution") and changes memory access settings.

Access by stopping execution **[E1] [E20] [EZ Emulator]**	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	Execution is temporarily halted before a read or write is performed.
		No	Memory is not accessed during execution.
Update display during the execution	Specify whether or not to update the displayed content of the Watch panel/Memory panel during program execution.		
	Default	*Yes*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	Display is updated during execution.
		No	Display is not updated during execution.
Display update interval[ms]	Specify the interval in 100 ms unit to update the displayed contents of the Watch panel or Memory panel while executing a program. Note that this property is displayed only when you selected [Yes] in the [Update display during the execution] property.		
	Default	*500*	
	How to change	By entering directly from the keyboard	
	Specifiable value	An integer in the range 100 to 65500 (Unit: fractions below 100 ms rounded up)	

Enable the automatic update of real-time display [E20(JTAG) [RX600, RX700 Series]]	Specify whether RRM area is automatically set. Note that this property is displayed only when [Real-time 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	Realtime display update is automatically set.
		No	Realtime display update is not set.

(3) [Register]
Change PC value display in the Status bar and display updating intervals during program execution.

PC display during the execution	Specify whether the PC value is displayed Note during program execution. **Caution** While current consumption is being measured, this property is fixed to [No].		
	Default	*No*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	PC value is displayed in the Status bar during execution.
		No	PC value is not displayed in the status bar during execution. "Running" is displayed in the Status bar.
Display update interval for PC[ms]	Specify an update interval in 100 ms units for the PC value which is displayed in the Status bar during program execution. Note that this property is displayed only when you've selected [Yes] in the [PC display during the execution] property.		
	Default	*500*	
	How to change	By entering directly from the keyboard.	
	Specifiable value	An integer in the range 100 to 65500 (Unit: fractions below 100 ms rounded up)	

Note [E1/E20/EZ Emulator [RX100 Series]]
PC values are hidden because these microcontrollers do not support display of the PC value in the status bar during program execution.

(4) [E2 Expansion Interface] [E2]
The detailed information on the E2 expansion interface is displayed and its configuration can be changed.

External trigger input	Make settings for the external trigger input. You can select different actions for each channel.	
Channel number (Subproperty)	The channel number is displayed.	
	Default	0 or 1
	Modifying	Changes not allowed

Use (Subproperty)	Specify whether to use the external trigger input for this channel number.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Uses the external trigger signal input through the selected channel.
		No	Does not use the external trigger signal input through the selected channel.
Input signal (Subproperty)	Specify the input signal.		
	Default	Rising Edge	
	Modifying	Select from the drop-down list.	
	Available values	Rising Edge	Detects rising edges of the external trigger signal input through the selected channel.
		Falling Edge	Detects falling edges of the external trigger signal input through the selected channel.
		Both Edges	Detects both rising and falling edges of the external trigger signal input through the selected channel.
		High	Detects the high level of the external trigger signal input through the selected channel.
		Low	Detects the low level of the external trigger signal input through the selected channel.
Action when inputting the external trigger (Subproperty)	The action when inputting the external trigger is displayed.		
	Default	Stop program	
	Modifying	Changes not allowed	
External trigger output	Set the settings related to the external trigger output. You can select different actions for each channel.		
Channel number (Subproperty)	The channel number is displayed.		
	Default	0 or 1	
	Modifying	Changes not allowed	
Use (Subproperty)	Specify whether to use the external trigger output for this channel number.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Uses the external trigger signal output through the selected channel.
		No	Does not use the external trigger signal output through the selected channel.
Output timing (Subproperty)	The output timing is displayed.		
	Default	Stop program	
	Modifying	Changes not allowed	

Output signal (Subproperty)	The output signal is displayed.	
	Default	High Pulse
	Modifying	Changes not allowed
Pulse width [us] (Subproperty)	Specify the pulse width.	
	Default	1
	Modifying	Directly enter from the keyboard.
	Available values	Integer number between 1 and 65535

(5) [Break] [E1] [E20] [EZ Emulator]
This category displays detailed information on the break function and changes break settings.

Type of breakpoints to be preferentially used	Specify the type of breakpoint that will be used first before the other when you set a breakpoint at the source line or execution address by clicking once with the mouse on the Editor panel/Disassemble panel.		
	Default	*Hardware break*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Software break	A software breakpoint is set first.
		Hardware break	A hardware breakpoint is set first.

(6) [System] [E1] [E20] [EZ Emulator]
Displays detailed information on emulation system or changes settings made in it.

Debug the program re-writing the on-chip PROGRAM ROM	Specify whether or not to debug a program that involves rewriting the on-chip program ROM (e.g., a program making use of ROM P/E mode).Note 1, Note 2		
	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 on-chip program ROM is debugged.
		No	A program that involves rewriting the on-chip program ROM is not debugged.
Debug the program re-writing the on-chip DATA FLASH	Specify whether or not to debug a program that involves rewriting the on-chip data flash (e.g., a program making use of data flash P/E mode).Note 1, Note 2, Note 4		
	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 on-chip data flash is debugged.
		No	A program that involves rewriting the on-chip 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.
Routine to run immediately before execution starts	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 the user program stops	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.
Routine to run immediately after execution stops	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 the user program stops] 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	
Work RAM start address for executing a specified routine	Specify the address where the work RAM for use in execution of the specified routine starts. Specify an address value that is a multiple of four bytes. If the entered value is not a multiple of four bytes, the value is automatically corrected. This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.		
	Default	*0x1000* Note 3	
	How to change	By entering directly from the keyboard However, changeable only when program execution is halted	
	Specifiable value	Address in the internal RAM of the microcontroller in use	

Work RAM size [bytes] for executing a specified routine	Indicates the size of the work RAM for use in execution of the specified routine. This property is displayed only when you have selected [Yes] either for the [Execute the specified routine immediately before execution of the user program] or [Execute the specified routine immediately after the user program stops] property.	
	Default	*Depends on selected microcontroller*
	How to change	Not changeable

Note 1. [E1] [E20] [EZ Emulator]
If you wish to use the Memory panel, for example, to reference data in the on-chip flash ROM (program ROM area) that has been programmed through the user program, select [Yes] for the [Debug the program rewriting the on-chip PROGRAM ROM] property. It is not possible to reference the latest values when this property is set to [No].
In the case of referencing data in the data flash area, also select [Yes] for the [Debug the program rewriting the on-chip PROGRAM ROM] property.

Note 2. [E1] [E20] [EZ Emulator]
While the user program is running, do not attempt to reference data (e.g. via the Memory panel) in the on-chip flash ROM (program ROM area or data flash area) that has been programmed through the user program. If this is attempted, incorrect values will be read out.

Note 3. The default value varies with the microcontroller.

Note 4. [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
When debugging a program that rewrites the option setting memory in the RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N groups, set the [Debug the program rewriting the on-chip DATA FLASH] property to [Yes].

(7) [Trace]
This category displays detailed information on the trace function and changes trace settings.

Caution [E20(JTAG) [RX600, RX700 Series]]
Part of the trace functions and real-time RAM monitor functions (RRM functions) can be used only on a mutually exclusive basis.

Usage of trace function [E1] [E20] [EZ Emulator]	Specify whether the trace function is used as real-time RAM monitor function (RRM function).		
	Default	*Trace*	
	How to change	- [E1(Serial)] [E1(JTAG)] [E20(Serial)] [EZ Emulator] 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.
		Real-time RAM monitor	Real-time RAM monitor function (RRM function) is used preferentially.
Use trace function [Simulator]	Specify whether or not 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	The trace function is used.
		No	The trace function is not used.

Clear trace memory before running [Simulator]	Specify whether or not to clear the trace memory before execution.		
	Default	*Yes*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The trace memory is cleared before execution.
		No	The trace memory is not cleared before execution.
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 over-write to trace memory	Continues overwriting trace data even after trace memory is used up.
		Stop trace [E1] [E20] [EZ Emulator]	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] [EZ Emulator]	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: - RX66x, RX65N-2M, RX651-2M, RX671, RX72T, RX72M, and RX72N Groups [E20] *Branch, Branch + Data access, Data access, Branch + Data access (without Access data), or Data access (without Access data)* - RX600, RX700 Series *Branch, Branch + Data access, or Data access* - RX100, RX200 Series *Branch or Data access*	
Start address for trace area without access data [E20][RX65N-2M, RX651-2M, RX66T, RX671, RX72T, RX72M, RX66N, and RX72N Group]	Specify the start address for the trace area without access data.		
	Default	*Start address on the internal RAM area*	
	How to change	By entering directly from the keyboard.	
	Specifiable value	Hexadecimal number in the range 0x0 to 0xFFFFFFFF	
End address for trace area without access data [E20][RX65N-2M, RX651-2M, RX66T, RX671, RX72T, RX72M, RX66N, and RX72N Group]	Specify the end address for the trace area without access data.		
	Default	*End address on the internal RAM area*	
	How to change	By entering directly from the keyboard.	
	Specifiable value	Hexadecimal number in the range 0x0 to 0xFFFFFFFF	

Bus Master of data access [E1/E20 [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]]	Specify the bus master which generated the data access. This property can be selected only when [Branch+Data access] or [Data access] is specified in the [Trace data type] property. When [Real-time RAM Monitor] is selected in the [Usage of trace function] property, this property is fixed to [CPU].		
	Default	*CPU*	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	CPU	The results of data access from CPU are displayed on the trace panel.
		DMAC/DTC	The results of data access from DMAC/DTC are displayed on the trace panel.
External trace output [E20(JTAG) [RX600, RX700 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, RX700 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	
Output timestamp [E1] [E20] [EZ Emulator]	Specify whether timestamp information is added to the collected trace data[Note 3]. This property is selectable only when you've specified [Trace] in the [Usage of trace function] property.		
	Default	*No*	
	How to change	By selecting from the drop-down list Changeable only when program execution is halted, however.	
	Specifiable value	Yes	Timestamp information is added.
		No	Timestamp information is not added.
Trace clock count source[MHz] [E1] [E20] [EZ Emulator]	Enter a count source with which a timestamp is calculated from a count value. However, this item is displayed only when you've selected [Yes] in the [Output timestamp] property.		
	Default	*Blank*	
	How to change	By entering directly from the keyboard.	
	Specifiable value	0.0001 to 999.999	

Division ratio of trace clock count source [E1/E20 [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]]	Specify the division ratio of trace clock count source for the timestamp. This property is displayed only when [Yes] is selected in the [Output timestamp] property. The frequency specified in the [Trace clock count source[MHz]] property is divided by the specified value (the frequency is multiplied by 1/n) and one cycle of the obtained frequency is used as the unit for timestamp count (the frequency for count value 1).		
	Default	*1/1*	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	1/1	The frequency of the timestamp count source is used without change.
		1/16	The frequency of the timestamp count source is multiplied by 1/16 before use.
		1/256	The frequency of the timestamp count source is multiplied by 1/256 before use.
		1/4096	The frequency of the timestamp count source is multiplied by 1/4096 before use.
Accumulate trace time [Simulator]	Specify a display method by which the trace time on the Trace panel is displayed.		
	Default	*No*	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	The trace time is displayed by an integral value.
		No	The trace time is displayed by a 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, 2M, 3M*	

Note 1. If [Start Tracing] or [Stop Tracing] on the context menu is selected on the Editor panel/Disassemble panel, this property is automatically changed to [Yes].

Note 2. The trade frame represents one unit of trace data.
One trace frame is used for a fetch, write, read, etc., respectively.

Note 3. This property is fixed to [No] when an RX100-series microcontroller is in use. Timestamp information is not included in trace data.

(8) [Timer] [E1] [E20] [EZ Emulator]
This category displays detailed information on the timer function and changes timer settings.

Use 64bit counter [RX600, RX700 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.

Caution RX100 Series other than RX140 Group does not support timers.

(9) [Coverage] [Simulator]

Use coverage function	Specify whether or not to use the coverage function.		
	Default	*No*	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	The coverage function is used.
		No	The coverage function is not used.
Reuse coverage result	Specify whether the coverage measurement result is loaded or saved when connected to or disconnected from the debug tool. This property is displayed only when you selected [Yes] in the [Use coverage function] property.		
	Default	*No*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The coverage measurement result is loaded or saved.
		No	The coverage measurement result is not loaded or saved.

(10) [Stream I/O] [Simulator]

Select stream I/O mode	Specify stream I/O mode.		
	Default	Simulator mode	
	How to change	By selecting from the drop-down list Changeable only when disconnected from the debug tool, however.	
	Specifiable value	Simulator mode	A simulator-inherent I/O method is used. Standard I/O and file I/O can be used.
		Emulator mode	An emulator-like I/O method is used. Only standard I/O can be used.
Use stream I/O function	Specify whether or not to use the stream I/O function. This property is displayed only when [Simulator mode] is specified in the [Select stream I/O mode] property.		
	Default	*No*	
	How to change	By selecting from the drop-down list However, changeable only when program execution is halted	
	Specifiable value	Yes	Stream I/O function is used.
		No	Stream I/O function is not used.
Stream I/O address	Specify the start position of a stream I/O that is used to perform standard I/O or file I/O from the user program. This property is displayed only when [Simulator mode] is specified in the [Select stream I/O mode] property.		
	Default	*0*	
	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	

(11) [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 [The microcontrollers with the FPU]	Specify whether or not to stop execution of the user program when a single-precision 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.

(12) [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.

(13) [Coverage] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]

Use code coverage function	Specify whether or not to use the code coverage function.		
	Default	*No*	
	How to change	By selecting from the drop-down list However, changeable only when CS+ is disconnected from the debug tool	
	Specifiable value	Yes	The code coverage function is used.
		No	The code coverage function is not used.

Coverage measurement priority	Specify the coverage measurement priority of the coverage data. This property is displayed only when [Yes] is specified in the [Use code coverage function] property.		
	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 takes priority. The coverage data may be lost during data output.
		Coverage measurement	Coverage measurement takes priority. CPU execution may stop to output coverage data, affecting real-time performance of program execution.
Coverage area of measurement	Specify the ranges where you want to measure coverage. The number of coverage area of measurement that can be registered is displayed as a main property. [Start address] and [Size of the coverage area of measurement [Mbytes]] are expanded to display in the sub-property. Specify a multiple of 4 Mbytes as the start address (if the specified value is not a multiple of 4M bytes, the value is automatically corrected). This property is displayed only when you selected [Yes] in the [Use coverage function] property.		
	Default	- Main item *4* - Sub-items <Start address> *Blank* <Size of the coverage area of measurement [Mbytes]> *4*	
	How to change	- Main item *Not changeable* - Sub-items <Start address> *By entering directly from the keyboard* <Size of the coverage area of measurement [Mbytes]> *Not changeable* *However, changeable only when program execution is halted*	
	Specifiable value	Hexadecimal number in the range 0x0 to 0xFFFFFFFF	
Reuse coverage result	Specify whether the coverage measurement result is loaded or saved when connected to or disconnected from the debug tool. This property is displayed only when you selected [Yes] in the [Use coverage function] property.		
	Default	*No*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The coverage measurement result is loaded or saved.
		No	The coverage measurement result is not loaded or saved.

[Download File Settings] tab

The [Download File Settings] tab displays detailed information for each category shown below and changes download file settings.
For details on how to download, see "2.5 Download and Upload."

(1) [Download]
(2) [Debug Information]

[Description of each category]

(1) [Download]
This category displays detailed information on download and changes download settings.

Download files	Specify a file to be downloaded[Note 1]. This property is expanded to display the file names to be downloaded and download conditions as a sub-property.		
	Default	[*Number of files to be downloaded*]	
	How to change	By selecting in the Download Files dialog box. The Download Files dialog box is opened by clicking the [...] button that is displayed at the right edge in the column when this property is selected. (No download files can be specified on the Property panel.)	
CPU Reset after download	Specify whether or not to reset the CPU after a download.		
	Default	*Yes*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The CPU is reset after a download.
		No	The CPU is not reset after a download.
Erase flash ROM before download [E1] [E20] [EZ Emulator]	Specify whether or not to erase the flash ROM (program ROM) before a download.		
	Default	*Yes*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The flash ROM is erased before a download.
		No	The flash ROM is not erased before a download.
Erase data flash ROM before down-load [E1] [E20] [EZ Emulator]	Specify whether or not to erase the data flash ROM before a download.		
	Default	*No*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The data flash ROM is erased before a download.
		No	The data flash ROM is not erased before a download.

Automatic change method of event setting position	Specify a method of how the set position (address) of a currently set event will be reset when it happens to be in the middle of an instruction as a result of reload[Note 2].		
	Default	*Suspend event*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Move to the head of instruction	The subject event is reset to the start address of an instruction.
		Suspend event	The subject event is held pending.

Note 1. The files specified to be the subject of a build process in the main project or a subproject cannot be removed from the subject files to be downloaded. (They are, by default, automatically registered as the download files.)
For details about the downloadable file formats, see "Table 2.2 Downloadable File Formats."

Note 2. This applies to only the event set position without debug information. If debug information is available, the event set position always moves to the beginning of a source text line.

(2) [Debug Information]
This category displays detailed information on debug information and changes debug settings.

Execute to the specified symbol after CPU Reset	Specify whether or not to execute the program up to a specified symbol position after the CPU is reset.		
	Default	*Yes*	
	How to change	By selecting from the drop-down list	
	Specifiable value	Yes	The program is executed up to a specified symbol position.
		No	The program is not executed after the CPU is reset.
Specified symbol	Specify the position at which the program executed after the reset of the CPU is halted. Note that this property is displayed only when you've selected [Yes] in the [Execute to the specified symbol after CPU Reset] property.		
	Default	*_main*	
	How to change	By entering directly from the keyboard	
	Specifiable value	0 to "*end address of address space*" address expression	
Specify the debugged overlay section	If the downloaded load module has overlay sections in it, select the section to debug.		
	Default	- 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	- [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. - File - Start address - End address - Priority section
The upper limit size of the memory usage[Mbytes]	Specify the maximum size [Mbytes] of memory to be used to read debug information. When an insufficient memory error occurs, specifying a smaller value for this maximum size may reduce the occurrence of the error. However, a smaller maximum size may degrade the response speed of the debug tool.	
	Default	*500*
	How to change	By entering directly from the keyboard
	Specifiable value	Integer between 100 and 1000

Caution 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.

Caution 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

The [Hook Transaction Settings] tab displays detailed information for each category shown below and changes hook process settings.
For details on how to set the hook process, see "2.18 Setting Up the Hook Process."

(1) [Hook Transaction Settings]

[Description of each category]

(1) [Hook Transaction Settings]
This category displays detailed information on hook process and changes hook process settings.
To set properties on this tab, use the Text Edit dialog box that is opened by clicking the [...] button that is displayed at the right edge in the column when you select any property. (Settings cannot be changed on this panel.)

Caution Up to 64 characters per process can be entered, and up to 128 processes per property can be specified. (One line in the [Text] area of the Text Edit dialog box corresponds to one process.)

Before download	Specify a process to be executed immediately before load module files are downloaded.	
	Default	*Before download [0]* (The value in "[ ]" denotes a currently specified number of processes.)
	How to change	By specifying in the Text Edit dialog box.
	Format	One of the following: - *I/O register name* + (single byte) space + *numeric value* [Process] The content of an I/O register is automatically rewritten with a numeric value. - *CPU register name* + (single byte) space + *numeric value* [Process] The content of the CPU register is automatically rewritten with a numeric value. - *Source* + (single byte) space + *Python script path* [Process] A script file specified by *Python script path* is executed.
After download	Specify a process to be executed immediately after load module files are downloaded.	
	Default	*After download [0]* (The value in "[ ]" denotes a currently specified number of processes.)
	How to change	By specifying in the Text Edit dialog box.
	Format	One of the following: - *I/O register name* + (single byte) space + *numeric value* [Process] The content of an I/O register is automatically rewritten with a numeric value. - *CPU register name* + (single byte) space + *numeric value* [Process] The content of the CPU register is automatically rewritten with a numeric value. - *Source* + (single byte) space + *Python script path* [Process] A script file specified by *Python script path* is executed.

After CPU reset under breaking	Specify a process to be executed immediately after the CPU is reset during a break.	
	Default	*After CPU reset under breaking [0]* (The value in "[ ]" denotes a currently specified number of processes.)
	How to change	By specifying in the Text Edit dialog box.
	Format	One of the following: - *I/O register name* + (single byte) space + *numeric value* [Process] The content of an I/O register is automatically rewritten with a numeric value. - *CPU register name* + (single byte) space + *numeric value* [Process] The content of the CPU register is automatically rewritten with a numeric value. - *Source* + (single byte) space + *Python script path* [Process] A script file specified by *Python script path* is executed.
Before running	Specify a process to be executed immediately before the program is started to run.	
	Default	*Before running* [0] (The value in "[ ]" denotes a currently specified number of processes.)
	How to change	By specifying in the Text Edit dialog box.
	Format	One of the following: - *I/O register name* + (single byte) space + *numeric value* [Process] The content of an I/O register is automatically rewritten with a numeric value. - *CPU register name* + (single byte) space + *numeric value* [Process] The content of the CPU register is automatically rewritten with a numeric value. - *Source* + (single byte) space + *Python script path* [Process] A script file specified by *Python script path* is executed.
After breaking	Specify a process to be executed immediately after the program execution is made to break.	
	Default	*After breaking[0]* (The value in "[ ]" denotes a currently specified number of processes.)
	How to change	By specifying in the Text Edit dialog box.
	Format	One of the following: - *I/O register name* + (single byte) space + *numeric value* [Process] The content of an I/O register is automatically rewritten with a numeric value. - *CPU register name* + (single byte) space + *numeric value* [Process] The content of the CPU register is automatically rewritten with a numeric value. - *Source* + (single byte) space + *Python script path* [Process] A script file specified by *Python script path* is executed.

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."

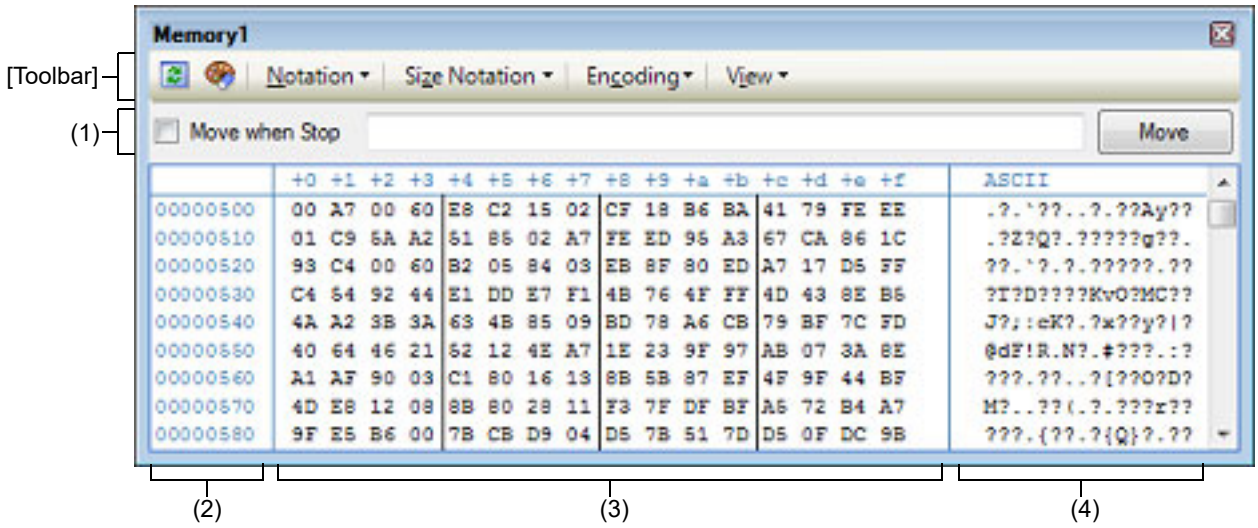
When memory values change after execution of the program, the display is automatically updated. (During step execution, the display is successively updated each time a step is executed.)

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 CS+ is connected with the debug tool.

Remark In a Scroll Range Settings dialog box that is opened by selecting [View] and then clicking the button in the toolbar, it is possible to set the scroll range of the vertical scroll bar of this panel.

Figure A.5 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.

Remark 1. By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.21.2 Symbol name completion function").

Remark 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] check box and the [Move] buttons are used to determine the timing with which the display start position is changed.

[Move when Stop]	☑	The address expression is automatically evaluated after the program has halted and the caret moves to the address derived from that calculation.
	☐	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] check box 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 display starts from address 0x0 by default.
However, an offset value of the start address can be set via the Address Offset Settings dialog box that is opened by selecting [Address Offset Value Settings...] from the context menu.
The address width matches that of the microcontroller's memory space specified in the project.
This area cannot be edited.

Caution The offset value that has been set is automatically changed in accordance with the number of view columns in the Memory value area.

(3) Memory value area
This area displays or changes memory values for the program mapped to the microcontroller.
To specify the display notation, display width of memory values, and the number of view columns, use the toolbar buttons or select [Notation], [Size Notation], and [View] from the context menu, respectively (see "2.11.1.2 Changing the display form of values").
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	
00 (Underlined)	Text color	Standard color	Memory values at addresses for which symbols are defined (Watch-expressions can be registered.) (See Register watch-expressions.)
	Background color	Standard color	
00	Text color	Sienna	Memory values that have changed as a result of program execution Note 1. Clicking the button in the toolbar resets the highlighting.
	Background color	LightYellow	
00	Text color	DeepPink	Memory values for which the Realtime display update function is enabled
	Background color	Standard color	

Example display (default)			Description	
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	PaleTur-quoise		
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 Note 2) 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		

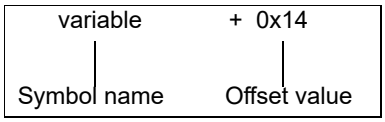
Note 1. Applicable only to the memory values in the address area that was displayed on the Memory panel immediately before the program execution. As the comparison is made before and after the program execution, highlighting will not be used if the value remained unchanged.

Note 2. [E1] [E20] [EZ Emulator]
The value of each I/O protection area is displayed as 00.

Caution The number of view columns is automatically changed in accordance with the set value of [Size Notation] of the context menu.

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., if the memory value is not underlined), no popup is displayed.



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 "2.11.1.4 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 "2.11.1.3 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 (see "2.11.1.5 Searching for memory contents").
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 (see "2.11.1.6 Collectively changing (initializing) memory contents").

(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 "2.11.1.7 Saving displayed memory contents".

(h) Zoom in or out on a view
To zoom in and out of the Memory panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Memory panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(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 (ASCII code is selected by default).
Furthermore, in this area, memory values converted into a floating-point value can be displayed as character strings. To do this, select the following item from [Encoding] of the context menu.

Item	Display Format		Size
Half-Precision Float	Half-precision floating-point value		16-bit
	Numeric value	*<sign><mantissa>*e*<sign><exponent>*	
	Infinite number	Inf, and -Inf	
	Not a number	NaN	
	Example	+ 1.234e+1	
Float	Single-precision floating-point value		32-bit
	Numeric value	*<sign><mantissa>*e*<sign><exponent>*	
	Infinite number	Inf, and -Inf	
	Not a number	NaN	
	Example	+ 1.234567e+123	
Double	Double-precision floating-point value		64-bit
	Numeric value	*<sign><mantissa>*e*<sign><exponent>*	
	Infinite number	Inf, and -Inf	
	Not a number	NaN	
	Example	+ 1.2345678901234e+123	
Float Complex	Complex number of single-precision floating-point		64-bit
	<Single-precision floating-point value> <Single-precision floating-point value> * I		
Double Complex	Complex number of double-precision floating-point		128-bit
	<Double-precision floating-point value> <Double-precision floating-point value> * I		
Float Imaginary	Imaginary number of single-precision floating-point		32-bit
	<Single-precision floating-point value> * I		
Double Imaginary	Imaginary number of double-precision floating-point		64-bit
	<Double-precision floating-point value> * I		

Caution Nothing is displayed when the minimum size of a character code or a floating-point value is greater than "the number of bytes of display width of memory values" x "the number of view columns".

This area has the following features.

(a) Edit character strings
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.)

Caution Character strings displayed as floating-point values cannot be edited.

(b) Searching character strings
The Memory Search dialog box is opened to search for character strings by selecting [Find...] from the context menu (see "2.11.1.5 Searching for memory contents").

(c) 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.
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 buttons that change the form in which memory value size is displayed. However, these buttons are 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 buttons that change the encoding in which character strings are displayed. However, these buttons are 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.
Half-Precision Float	Displays character strings as a half-precision floating-point value.
Float	Displays character strings as a single-precision floating-point value.
Double	Displays character strings as a double-precision floating-point value.
Float Complex	Displays character strings as a complex number of single-precision floating-point.
Double Complex	Displays character strings as a complex number of double-precision floating-point.
Float Imaginary	Displays character strings as an imaginary number of single-precision floating-point.
Double Imaginary	Displays character strings as an imaginary number of double-precision floating-point.
View	Shows the following buttons that change the display form.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range.
Column Number Settings...	Opens the Column Number Settings dialog box to set the number of view columns in the Memory value area.
Address Offset Value Settings...	Opens the Address Offset Settings dialog box to set an address and an offset value for addresses displayed in the Address area.

[[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 "(c) 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 (unless the floating-point value display is selected) 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 "(c) 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 display form 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.
Half-Precision Float	Displays character strings as a half-precision floating-point value.
Float	Displays character strings as a single-precision floating-point value.
Double	Displays character strings as a double-precision floating-point value.
Float Complex	Displays character strings as a complex number of single-precision floating-point.
Double Complex	Displays character strings as a complex number of double-precision floating-point.
Float Imaginary	Displays character strings as an imaginary number of single-precision floating-point.
Double Imaginary	Displays character strings as an imaginary number of double-precision floating-point.
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.
Column Number Settings...	Opens the Column Number Settings dialog box to set the number of view columns in the Memory value area.
Address Offset Value Settings...	Opens the Address Offset Settings dialog box to set an offset value for addresses displayed in the Address area.
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 displays the disassembled results of memory contents (disassembled text), as well as displays the results of line assemble (see "2.6.4 Performing line assembly"), instruction-level debugging (see "2.9.3 Execute programs in steps"), and code coverage measurements [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]] (see "2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]").

Up to four instances of this panel can be displayed at a time. Each panel is discriminated by the name "Disassemble1," "Disassemble2," "Disassemble3," and "Disassemble4" in the title bar.

By displaying this panel in mixed display mode, it is also possible to display the source text in a source file corresponding to code data (default).

Note that this panel can only be opened when Cube Suite+ is connected with the debug tool.

Caution If a program is stepped through while the focus exists on this panel, the program is executed one instruction at a time. (See "2.9.3 Execute programs in steps.")

Remark 1. In a Scroll Range Settings dialog box that is opened by selecting [View] and then clicking the button in the toolbar, it is possible to set the scroll range of the vertical scroll bar of this panel.

Remark 2. By choosing [Print...] from the [File] menu, it is possible to print the picture image currently being displayed on this panel.

Figure A.6 Disassemble Panel (When Displayed in Mixed Display Mode)

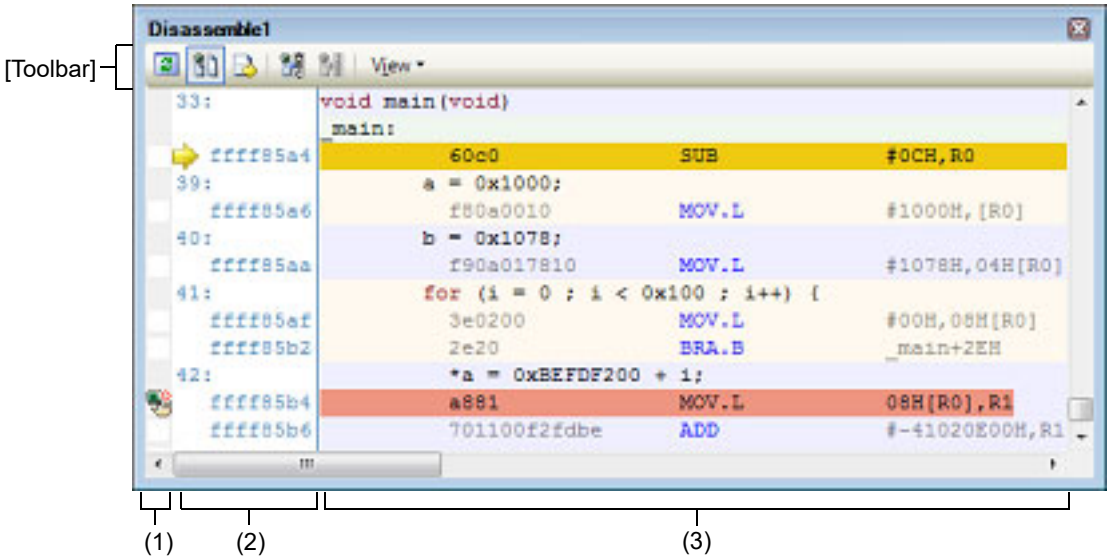


Figure A.7 Disassemble Panel (When Mixed Display Mode is Turned Off)

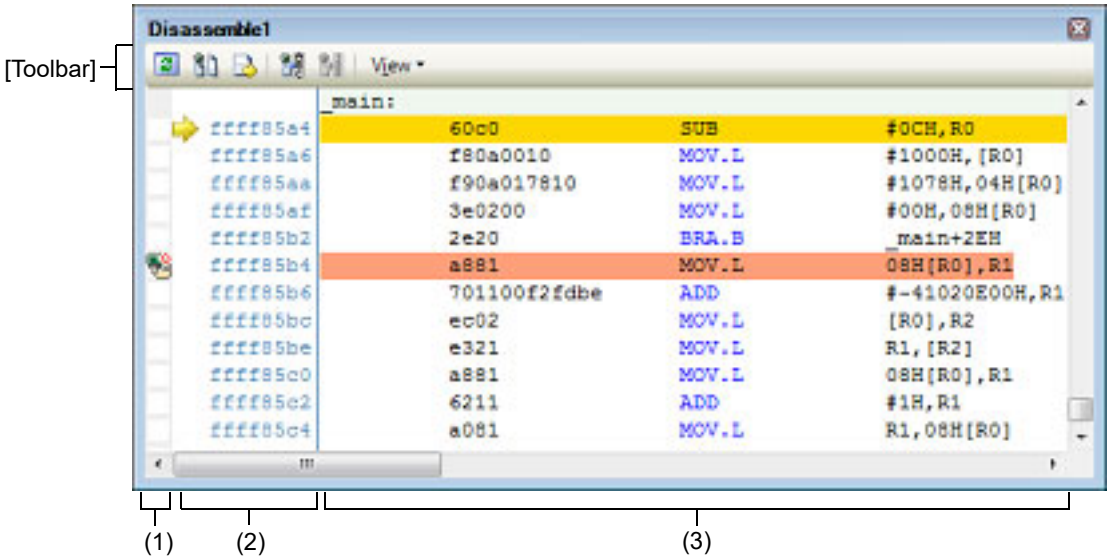
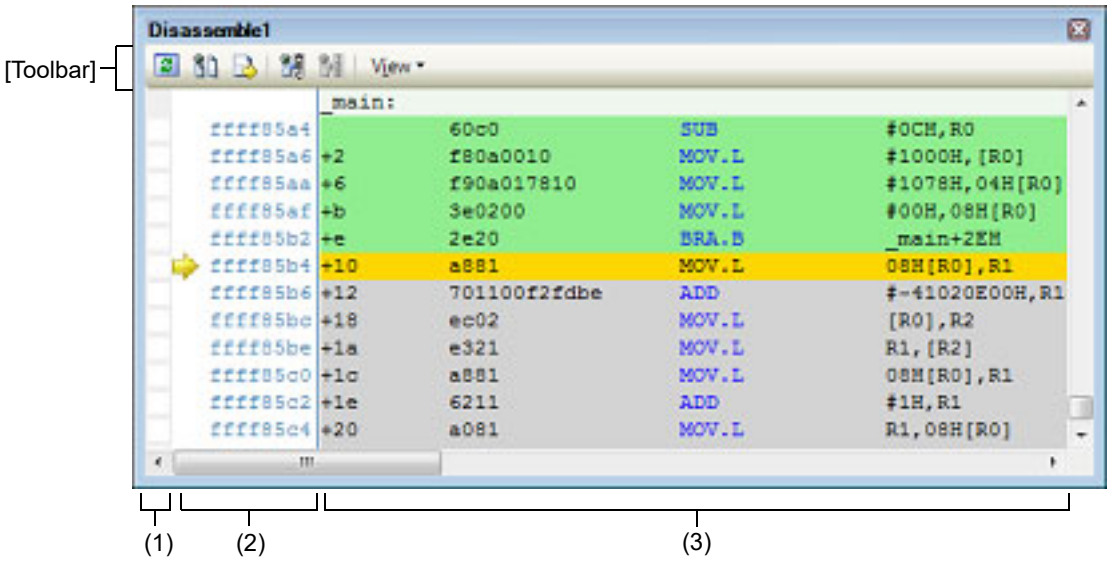


Figure A.8 Disassemble Panel (When Code Coverage Measurement Result is Displayed) [Simulator][E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Disassemble Panel-Only Items)]
- [[Edit] menu (Disassemble Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Disassemble] from the [View] menu and then select [Disassemble*1* - *4*]

[Description of each area]

(1) Event area
The lines where an event can be set are displayed against a white background. (If a line is displayed against a gray background, it means that no events can be set on that line.)
If there is a currently set event on a line, an Event mark denoting the type of event is displayed on the line.
This area has the following features:

(a) Setting/clearing a breakpoint
Click a place with the mouse where you want to set a breakpoint. That way, it is possible to set a breakpoint easily.
A breakpoint is set in an instruction at the beginning address corresponding to the line position you've clicked.
When a breakpoint is set, an Event mark is displayed on the line where you've set the breakpoint. Also, detailed information on the set breakpoint is reflected on the Events panel.
Note that if you've done this operation at a place where any event mark is already displayed, the event is deleted and no breakpoint is set.
Also note that an event can be set on only a line that is displayed against a white background.
For details on how to set a breakpoint, see "2.10.2 Stop the program at the arbitrary position (breakpoint)."

(b) Altering states of various events
Right-clicking any event mark brings up the menu shown below, allowing you to change the state of a selected event.

Enable Event	Changes the state of a selected event to an Enabled state. When a specified condition is true, the event concerned occurs. Note that if the event mark denoting that there are multiple events set is selected, all of the set events are enabled.
Disable Event	Changes the state of a selected event to a Disabled state. Even when a specified condition is true, the event concerned does not occur. Note that if the event mark denoting that there are multiple events set is selected, all of the set events are disabled.
Delete Event	Deletes a selected event. Note that if the event mark denoting that there are multiple events set is selected, all of the set events are deleted.
View Details in Event Panel	Opens an Events panel that shows detailed information on a selected event.

(c) Popup display
When the mouse cursor is hovered over an Event mark, the event name of the event, detailed information on it, and the comment added to the event are displayed in a popup box.
Note that if there are multiple events set at the relevant place, information on each event, for up to three, is enumerated.

(2) Address area
This area displays the beginning address of disassembly on each line (always displayed in hexadecimal).
Also, the current PC mark ( ) denoting the current PC position (PC register value) is displayed.
The address width is equal to that of the memory space of the microcontroller that is specified in the project.
Note that, for the source text line during mixed display mode, a line number (*xxx*:) in a source file corresponding to the beginning address is displayed.
This area has the following features:

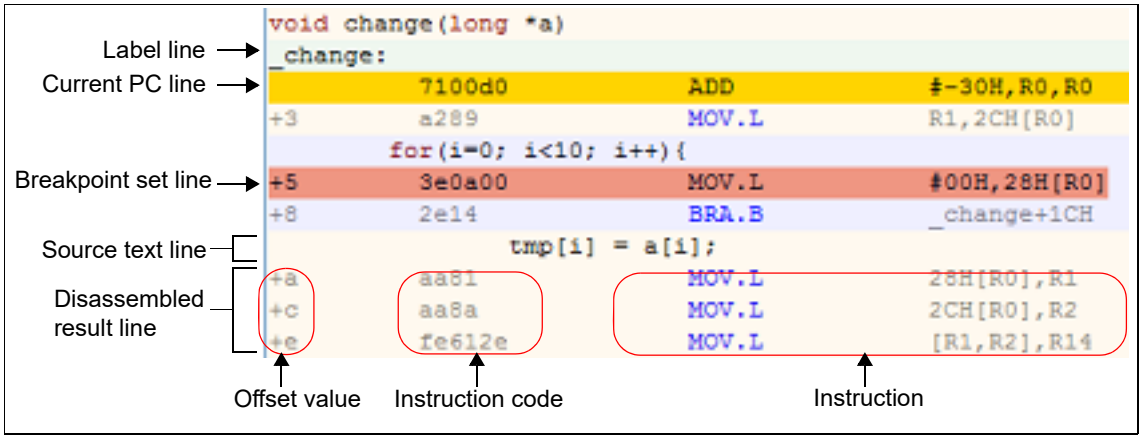
(a) Popup display
When the mouse cursor is hovered over an address/source line number, the following information is displayed in a popup box.

Address	Format: *<load module name>*[Note]$*<label name>* + *<offset value>* Example1: test1.out$main + 0x10 Example2: sub function + 0x20
Source line number	Format: *<load module name>*[Note]$*<file name>* # *<line number>* Example1: test1.out$main.c#40 Example2: main.c#100

Note The load module name is displayed only when there are multiple load module files that have been downloaded.

(3) Disassemble area
Following the source text line concerned, a disassembled result line is displayed as shown below.

Figure A.9 Contents of Displayed Disassemble Area (When Displayed in Mixed Display Mode)



Label line		Shows a label name, if a label is defined at the address, and highlights the entire line in pale green color.
Current PC line		Highlights a line corresponding to the address of the current PC position (PC register value).[Note 1]
Breakpoint set line		Highlights a line where a breakpoint is set.[Note 1]
Source text line		Shows source text corresponding to code data.[Note 2]
Disassembled result line	Offset value	Shows an offset value from the nearest label, if no labels are defined at the address.[Note 3]
	Instruction code	Shows the code that was disassembled in hexadecimal.
	Instruction	Shows an instruction resulting from disassembly. The mnemonic is highlighted in blue.

Note 1. The highlighting color depends on how the [General - Font and Color] category in the Option dialog box are set.

Note 2. By clicking the button (toggle) in the toolbar or unchecking [Mixed Display] on the context menu, it is possible to hide the source text. (By default, this menu item is checked.)

Note 3. The offset value is, by default, not displayed. To show it, click the button in the toolbar or select [Show Offset] on the context menu.

This area has the following features:

(a) Line assembly
The displayed instruction/code can be edited (assembled in-line).
For details on how to do it, see "2.6.4 Performing line assembly."

(b) Program execution at instruction level
By stepping through a program while the focus exists on this panel, it is possible to control program execution at the instruction level.
For details on how to do it, see "2.9.3 Execute programs in steps."

(c) Setting various events
By selecting [Break Settings], [Trace Settings] or [Timer Settings] on the context menu, it is possible to set various events at the address or line at which the caret currently exits.
When an event is set, the corresponding Event mark is displayed in the Event area. Also, detailed information on the set event is reflected on the Events panel.
However, events can only be set on a line that is displayed against a white background in the event area.
For details on how to set an event, see the following sections:

- "2.10.3 Stop the program at the arbitrary position (break event) [E1] [E20] [EZ Emulator]"
- "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"

Remark Breakpoints can also be set or cleared easily in the Event area. (See "(a) Setting/clearing a breakpoint.")

(d) Registering a watch-expression
The displayed C variable, CPU register, I/O register or assembler symbol can be registered as a watch-expression in the Watch panel.
For details on how to do it, see "2.11.6.1 Registering watch-expressions."

(e) Moving to a symbol defined location
While the caret has been moved to an instruction that references a symbol, click the button in the toolbar or select [Go to Symbol] on the context menu. The caret position is moved to the address at which the symbol at the caret position is defined.

Also, following this operation, click the button in the toolbar or select [Back to *Address*] on the context menu. The caret position is returned to the instruction that was referencing the symbol before the caret was moved. (*Address* denotes the address value of the instruction that references the symbol.)

(f) Jumping to a source line/memory value
When [Jump to Source] on the context menu is selected, the Editor panel is opened, with the caret on it moved to a source line corresponding to the address of the current caret position. (If the Editor panel is already open, CS+ jumps to it directly.)
Similarly, when [Jump to Memory] on the context menu is selected, the Memory panel (Memory1) is opened, with the caret on it moved to a memory value corresponding to the address of the current caret position. (If the Memory panel is already open, CS+ jumps to it directly.)

(g) Displaying a code coverage measurement result [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]
If the coverage function is enabled, the lines included in the area that was subjected to coverage measurement are highlighted based on the code coverage measurement result acquired 2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]] by program execution.
For details about coverage measurement, see "2.15 Measure Coverage [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]."

(h) Saving disassembled data
By choosing [Save Disassemble Data As…] from the [File] menu, it is possible to open the Data Save dialog box and save the content of this panel in a text file (*.txt) or CSV file (*.csv).
For details on how to save disassembled data, see "2.6.2.5 Saving the displayed contents of disassembled results."

(i) Zoom in or out on a view
To zoom in and out of the Disassemble panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Disassemble panel.
You can also change the zoom ratio 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. 25%).

Remark The following items can be customized by setting the Option dialog box.

- Display fonts
- Colors of reserved words/comments

[Toolbar]

	Gets latest information from the debug tool to update the display.
	Selects mixed display mode that shows the correspondence between disassembled result and source text (default).
	Specifies that the caret position track the current PC value.
	Moves the caret to the position at which a selected symbol is defined.
	Moves the caret to a position (*address*) at which it was immediately before being moved with the button.
View	Shows the following buttons that change the form in which a disassemble area is displayed.
Show Offset	Displays the offset value of a label. If no labels are defined at the address, the offset value from the nearest label is displayed.
Show Symbol	Displays an address value in the form "symbol + offset value" (default). However, if the address value has a symbol defined in it, only the symbol is displayed.
	Opens the Scroll Range Settings dialog box to set a scroll range.

[[File] menu (Disassemble Panel-Only Items)]

The [File] menu used exclusively for the Disassemble panel is as follows. (The other items are shared.)
However, all of these items are disabled during program execution.

Save Disassemble Data	Saves the disassembled content to a text file (*.txt) or CSV file (*.csv) that has been saved previously (see "(h) Saving disassembled data"). Note that if this item is selected for the first time after startup, the same operation as [Save Disassemble Data As…] would have been selected is performed.
Save Disassemble Data As…	Opens the Data Save dialog box to save the disassembled content to a specified text file (*.txt) or CSV file (*.csv) (see "(h) Saving disassembled data").
Print...	Opens the Print Address Range Settings dialog box to print the content of this panel.

[[Edit] menu (Disassemble Panel-Only Items)]

The [Edit] menu used exclusively for the Disassemble panel is as follows. (All other items are disabled.).

Copy	Copies the content of a selected line as a character string to the clipboard, if a line is selected. When in edit mode, a selected character string is copied to the clipboard.
Rename	Goes to edit mode to edit the instruction or instruction code at the caret position (see "2.6.4 Performing line assembly"). However, this is disabled during program execution.
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.
Go To…	Opens the Go to the Location dialog box to move the caret to a specified address.

[Context menu]

[Disassemble area and Address area]

Register to Watch1	Registers a selected character string or a word at the caret position as a watch-expression in the Watch panel (Watch1). (Word determination depends on the current build tool.) When registered as a watch-expression, they are registered as a variable name, so that the displayed symbol name varies with scope.
Register Action Event...	Opens the Action Events dialog box to set an action event at an address at the caret position.
Go to Here	Executes the program from an address indicated by the current PC value to an address corresponding to the line where the caret is positioned. However, this is disabled during program execution or when [Build & Download] is under execution.
Set PC to Here	Changes the current PC value to an address of the line where the caret currently exists. However, this is disabled during program execution or when [Build & Download] is under execution.
Go To…	Opens the Go to the Location dialog box to move the caret to a specified address.
Go to Symbol	Moves the caret to the position where a selected symbol is defined.
Back to *Address*	Moves the caret to a position (*address*) at which it was immediately before being moved by [Go to Symbol]. However, this is disabled when no symbol names are displayed at address.

Break Settings	Shows the following cascaded menu to set break-related events. Note that events can only be set on event settable lines (see "(1) Event area").
Set Hardware Break	Sets a breakpoint (hardware break event) at the address where the caret is positioned (see "2.10.2 Stop the program at the arbitrary position (breakpoint)."
Set Software Break [E1] [E20] [EZ Emulator]	Sets a breakpoint (software break event) at the address where the caret is positioned (see "2.10.2 Stop the program at the arbitrary position (breakpoint)."
Set Read Break to [Simulator]	Sets a break event for read access at the caret position or in a selected variable (global variable, static variable inside a function or static variable inside a file) or an I/O register (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Set Write Break to [Simulator]	Sets a break event for write access at the caret position or in a selected variable (global variable, static variable inside a function or static variable inside a file) or an I/O register (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Set R/W Break to [Simulator]	Sets a break event for read/write access at the caret position or in a selected variable (global variable, static variable inside a function or static variable inside a file) or an I/O register (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Set Combination Break [E1] [E20] [EZ Emulator]	Sets a break at the caret position's address or the caret position or a selected variable (global variable, static variable inside a function, static variable inside a file) or I/O register as the condition for a combination break event (see "2.10.3.1 Set a beak event (execution-related)").
Set Read Combination Break to [E1] [E20] [EZ Emulator]	Sets a break event by a read access to the caret position or a selected variable (global variable, static variable inside a function, static variable inside a file) or I/O register as the condition for a combination break (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Set Write Combination Break to [E1] [E20] [EZ Emulator]	Sets a break event by a write access to the caret position or a selected variable (global variable, static variable inside a function, static variable inside a file) or I/O register as the condition for a combination break (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Set R/W Combination Break to [E1] [E20] [EZ Emulator]	Sets a break event by a read/write access to the caret position or a selected variable (global variable, static variable inside a function, static variable inside a file) or I/O register as the condition for a combination break (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Break Option	Opens the Property panel to set a break function.

Trace Settings	Shows the following cascaded menu to set a trace-related event. Note that events can only be set on event settable lines (see "(1) Event area").
Start Tracing	Sets a trace start event that causes collection of trace data indicating a program execution history to start when an instruction at the address where the caret exists is executed (see "2.13.3.1 Setting a trace start event and a trace end event"). [Simulator] The setting of the [Use trace function] property in the [Trace] category on the Property panel is automatically set to [Yes].
Stop Tracing	Sets a trace end event that causes collection of trace data indicating a program execution history to end when an instruction at the address where the caret exists is executed (see "2.13.3.1 Setting a trace start event and a trace end event"). [Simulator] The setting 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 that, when the caret position or a selected variable (global variable, static variable inside a function or static variable inside a file) or an I/O register is accessed for read, causes the accessed value to be recorded in trace memory (see "(1) When an access to a variable or I/O register occurred").
Record Writing Value	Sets a point trace event that, when the caret position or a selected variable (global variable, static variable inside a function or static variable inside a file) or an I/O register is accessed for write, causes the accessed value to be recorded in trace memory (see "(1) When an access to a variable or I/O register occurred").
Record R/W Value	Sets a point trace event that, when the caret position or a selected variable (global variable, static variable inside a function or static variable inside a file) or an I/O register is accessed for read/write, causes the accessed value to be recorded in trace memory (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 inside a function, static variable inside a file) or I/O register (see "2.13.3.1 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 inside a function, static variable inside a file) or I/O register (see "2.13.3.1 Setting a trace start event and a trace end event").
Show Trace Result	Opens the Trace panel and displays acquired trace data.
Trace Settings	Opens the Property panel to set the trace function. However, this is disabled while the tracer is operating.

Timer Settings[Note 1]	Shows the following cascaded menu to set a timer-related event (see "2.14.3 Measuring execution time in a section"). Note that events can only be set on event settable lines (see "(1) Event area").
Start Timer	Sets a timer start event that causes measurement of program execution time to start when an instruction at the address where the caret exists is executed (see "(1) How to set a timer start event").
Stop Timer	Sets a timer end event that causes measurement of program execution time to end when an instruction at the address where the caret exists is executed (see "(2) 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 inside a function, static variable inside a file) or I/O register (see "(1) How to set a timer start event").
Set Timer <N> [E1] [E20] [EZ Emulator]	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 inside a function, static variable inside a file) or I/O register (see "(2) How to set a timer end event").
Set Timer <N> [E1] [E20] [EZ Emulator]	Specify a channel[Note 2] in which a timer start event is set.
View Result of Timer	Opens the Events panel to display only timer-related events.
Clear Coverage Information [Simulator] [E20 [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]]	Clears all of the code coverage measurement results that the debug tool currently holds. However, this item is not displayed if the debug tool used does not support the coverage function.
Edit Disassemble	Goes to edit mode to edit an instruction at the line where the caret is positioned (see "2.6.4 Performing line assembly"). However, this is disabled during program execution.
Edit Code	Goes to edit mode to edit an instruction code at the line where the caret is positioned (see "2.6.4 Performing line assembly"). However, this is disabled during program execution.
View	Shows the following cascaded menu to set the contents displayed in the disassemble area.
Show Offset	Displays the offset value of a label. If no labels are defined at the address, the offset value from the nearest label is displayed.
Show Symbol	Displays an address value in the form "symbol + offset value" (default). However, if the address value has a symbol defined in it, only the symbol is displayed.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set a scroll range.
Mixed Display	Selects mixed display mode that shows the correspondence between disassembled result and source text (default).
Jump to Source	Opens the Editor panel, with the caret on it moved to a source line corresponding to the address at the caret position.
Jump to Memory	Opens the Memory panel (Memory1), with the caret on it moved to a memory value corresponding to the address at the caret position.

Note 1. [E1] [E20] [EZ Emulator]
RX100 Series other than RX140 Group does not support timers.

Note 2. The specifiable number of channels differs between the RX600/RX700 and RX200 Series/RX140 Group, as shown below.
RX600, RX700 Series: 2 (32 bits * 2) or 1 (64 bits * 1)
RX200 Series, RX140 Group: 1 (24 bits * 1)

[Event area] (**[E1] [E20] [EZ Emulator]**)

Hardware Break First	Defines the type of break that can be set by clicking the mouse once to be a hardware breakpoint (reflected in the setting of the [Type of breakpoints to be preferentially used] in the [Break] [E1] [E20] [EZ Emulator] category on the Property panel.
Software Break First	Defines the type of break that can be set by clicking the mouse once to be a software breakpoint (reflected in the setting of the [Type of breakpoints to be preferentially used] in the [Break] [E1] [E20] [EZ Emulator] category on the Property panel).

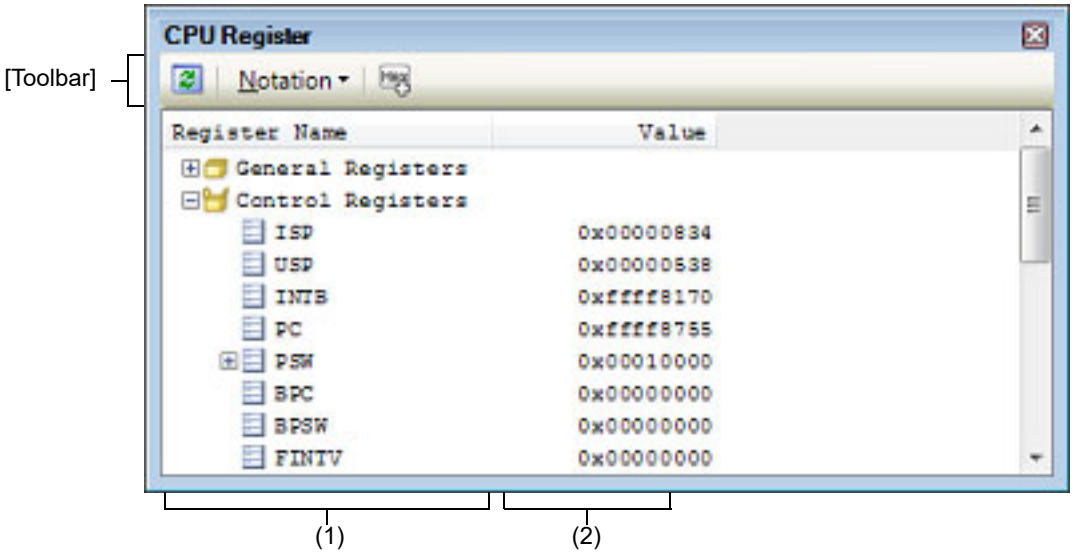
CPU Register panel

This panel displays the contents of the CPU registers (general-purpose and control registers) and change register values (see "2.11.2 Displaying and changing the CPU registers")

Note that this panel can be opened only when CS+ is connected with the debug tool.

Remark By double-clicking a line delimiting each area on the panel, it is possible to change the relevant area to the smallest displayable width without omitting the content in it.

Figure A.10 CPU Register Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] Menu (CPU Register Panel-Only Items)]
- [[Edit] Menu (CPU Register Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [CPU Register] from the [View] menu.

[Description of each area]

(1) [Register Name] area

This area displays register names in list form, with the types of registers classified by category (folder).
The meaning of each icon displayed here is described below.
Note that the category names or register names displayed here cannot be edited nor deleted.

	Indicates that the register names belonging to this category are displayed. Double-clicking the icon or clicking the "-" mark closes the category and hides the register names.
	Indicates that the register names belonging to this category are hidden. Double-clicking the icon or clicking the "+" mark opens the category and displays register names.
	Shows a register name. Double-clicking the icon or clicking the "+" or "-" mark displays or hides the low-level register names (names representing part of a register).

	Shows a register name (name representing part of a register).

The displayed category and register names are as follows. (The number of the "+" marks at the beginning of each register name denotes the depth of the hierarchical level of displayed registers.)

Table A.2 [General Registers] Category and Register Names [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 [Double-Precision Floating-Point Data Registers] Category and Register Names [RX][Note 5]

Register name (alias)	Bit width	Register name (alias)	Bit width
+ DR0	64	+ DR8	64
+ DR1	64	+ DR9	64
+ DR2	64	+ DR10	64
+ DR3	64	+ DR11	64
+ DR4	64	+ DR12	64
+ DR5	64	+ DR13	64
+ DR6	64	+ DR14	64
+ DR7	64	+ DR15	64

Table A.4 [Control Registers] Category and Register Names [RX]

Register name (alias)	Bit width	Register name (alias)	Bit width	Register name (alias)	Bit width
+ ISP	32	+ FPSW[Note 2]	32	+ ACC[Note 3]	64
+ USP	32	++ FS	1	+ ACC0[Note 4]	72
+ INTB	32	++ FX	1	+ ACC1[Note 4]	72
+ PC	32	++ FU	1	+ EXTB[Note 4]	32
+ PSW	32	++ FZ	1	+ DPSW[Note 5]	32
++ IPL	4[Note 1]	++ FO	1	+ DCMR[Note 5]	32
++ PM	1	++ FV	1	+ DECNT[Note 5]	32
++ U	1	++ EX	1	+ DEPC[Note 5]	32
++ 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		
		++ CY	1		
		++ RM	2		

Note 1. The bit width is 3 bits for the RX610 Group.

Note 2. FPSW register is not supported by the microcontrollers without the FPU.

Note 3. The RX71M, RX66x, RX65x, RX64M, RX671, RX24x, RX23x, RX72T, RX72M, and RX72N groups do not support the ACC register.

Note 4. Only the RX71M, RX66x, RX65x, RX64M, RX671, RX24x, RX23x, RX72T, RX72M, and RX72N groups support the ACC0, ACC1, and EXTB registers.

Note 5. These registers are not supported in microcontrollers that do not incorporate a double-precision floating-point coprocessor.

This area has the following features.

(a) Registration of watch-expressions
CPU registers or categories can be registered as watch-expressions in the Watch panel.
For details on how to do it, see "2.11.6.1 Registering watch-expressions."

Remark 1. If watch-expressions are registered for a category as the subject of registration, all CPU registers belonging to the category are registered as watch-expressions.

Remark 2. The registered watch-expressions will have scope specification automatically added.

Caution [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
When the ACC0 or ACC1 register is registered as a watch-expression in the Watch panel and

[Float] or [Double] is selected as the value display form, the actual display is the same as when [Hexadecimal] is selected.

(2) [Value] area

This area displays the value of each CPU register and changes register values.
The desired notation (numerical representation) can be selected by clicking the appropriate toolbar button or by selecting from a 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 CPU 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	Values of CPU registers which have had their values changed by the user (Written into the target memory by hitting the [Enter] key)
	Background color	Standard color	
0x0	Text color	Sienna	Values of CPU registers whose values have changed as a result of program execution Highlighting is reset by reexecuting the program.
	Background color	LightYellow	

This area has the following features:

(a) Alternation of CPU register values

To change the value of a CPU register value, select the CPU register value you want to change and click it again, and then enter a new value directly from the keyboard. (Pressing the [Esc] key cancels the editing mode.)
When you've finished editing the value of a CPU register, press the [Enter] key or move the focus to other than the edit area. The value you've changed is written into the register of the debug tool.

(b) Saving of CPU register values

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 entire content of this panel to a text file (*.txt) or CSV file (*.csv).
For details on how to save the CPU register values, see "2.11.2.3 Saving the displayed CPU register contents."

(c) Zoom in or out on a view

To zoom in and out of the CPU Register panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the CPU Register panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

[Toolbar]

	Acquires latest information from the debug tool to update the display. However, this is disabled during program execution.
Notation	Shows the following buttons for changing the form in which values are displayed.
AutoSelect	Displays the value of a selected item (including a low-level item) in predetermined notation (default).
Hexadecimal	Displays the value of a selected item (including a low-level item) in hexadecimal.
Signed Decimal	Displays the value of a selected item (including a low-level item) in signed decimal.
Unsigned Decimal	Displays the value of a selected item (including a low-level item) in unsigned decimal.
Octal	Displays the value of a selected item (including a low-level item) in octal.
Binary	Displays the value of a selected item (including a low-level item) in binary.
ASCII	Displays the character string of a selected item (including a low-level item) in ASCII code. If the subject to be displayed consists of 2 bytes or more, characters of 1 byte each are displayed in a row.
Float	Displays a selected item in Float. Except for 4-byte data, however, they are displayed in predetermined notation.
Double	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 "( )".

[[File] Menu (CPU Register Panel-Only Items)]

The [File] menu used exclusively for the CPU Register panel is as follows. (The other items are shared.)
However, all of these items are disabled during program execution.

Save CPU Register 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 CPU register values"). If this item is selected for the first time after startup, the same operation as [Save CPU Register Data As…] would have been selected is performed.
Save CPU Register 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 CPU register values").

[[Edit] Menu (CPU Register Panel-Only Items)]

The [Edit] menu used exclusively for the CPU Register panel is as follows. (All other items are disabled.)

Cut	Cuts a selected range of character string and copies it to the clipboard. However, this is enabled only when character strings are being edited.
Copy	When in editing mode, copies a selected character string to the clipboard. If a line is selected, a register or category on the line is copied to the clipboard. Note that a copied item can be pasted to the Watch panel.
Paste	Pastes a copied character string from the clipboard to the caret position. However, this is enabled only when character strings are being edited.
Select All	Selects all items.
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 a selected register name or category to the Watch panel (Watch1).
Copy	When in editing mode, copies a selected character string to the clipboard. If a line is selected, a register item or category on the line is copied to the clipboard. Note that a copied item can be pasted to the Watch panel.
Notation	Shows the following cascaded menu to specify the form in which values are displayed.
AutoSelect	Displays the value of a selected item (including a low-level item) in predetermined notation (default).
Hexadecimal	Displays the value of a selected item (including a low-level item) in hexadecimal.
Signed Decimal	Displays the value of a selected item (including a low-level item) in signed decimal.
Unsigned Decimal	Displays the value of a selected item (including a low-level item) in unsigned decimal.
Octal	Displays the value of a selected item (including a low-level item) in octal.
Binary	Displays the value of a selected item (including a low-level item) in binary.
ASCII	Displays the character string of a selected item (including a low-level item) in ASCII code. If the subject to be displayed consists of 2 bytes or more, characters of 1 byte each are displayed in a row.
Float	Displays a selected item in Float. Except for 4-byte data, however, they are displayed in predetermined notation.
Double	Displays a selected item in Double. Except for 8-byte data, however, they are displayed in predetermined notation.
Include Hexadecimal Value	Adds a hexadecimal equivalent for the displayed value at the end of it, with the equivalent enclosed in parentheses "( )".

Caution [RX71M, RX66x, RX65x, RX64M, RX671, RX72T, RX72M, and RX72N Groups]
When [Float] or [Double] is selected for the ACC0 or ACC1 register, the value which is displayed is the same as when [Hexadecimal] is selected.

IOR panel

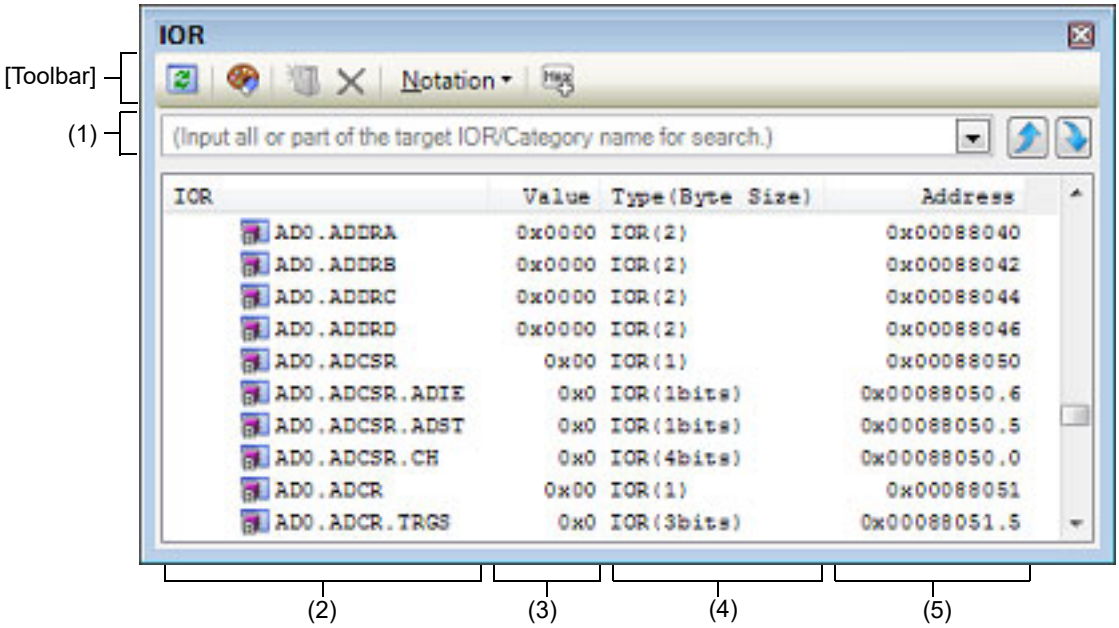
This panel displays the contents of I/O registers and changes their values (see Section"2.11.3 Displaying and changing the I/O registers").

Note that this panel can only be opened when CS+ 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 [Force Read Value] 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.11 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.

Remark 1. The I/O register names that are hidden as classified by category (folder) also are searched (expanded and placed in selected state).

Remark 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.

Caution 1. A category cannot be created within another category.

Caution 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 "2.11.6.1 Registering watch-expressions".

Remark 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.

Remark 2. The registered watch-expressions have their scope specification automatically given.

(c) Zoom in or out on a view
To zoom in and out of the IOR panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the IOR panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(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	Sienna	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	LightYellow	
?	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 "2.11.3.4 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 "2.11.3.6 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.	

Example 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.

Example 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.

Example 1. For the "0xFF40" case
This register is mapped to "0xFF40".

Example 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.
Hexadecimal	Displays the value of a selected item in hexadecimal (default).
Signed Decimal	Displays the value of a selected item in signed decimal.
Unsigned Decimal	Displays the value of a selected item in unsigned decimal.
Octal	Displays the value of a selected item in octal.
Binary	Displays the value of a selected item in binary.
ASCII	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.
Go To...	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").

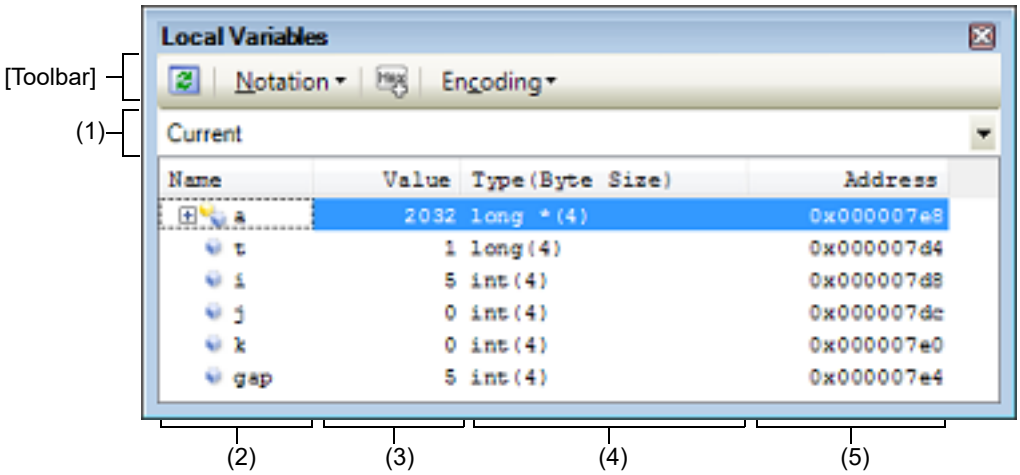
Note that this panel can only be opened when CS+ is connected with the debug tool.

Caution 1. During program execution, nothing is displayed on this panel. The contents of each area on this panel are displayed at the time the program has stopped running.

Caution 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.12 Local Variables panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Local Variables Panel-Only Items)]
- [[Edit] menu (Local Variables Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Local Variable] 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.

Item	Operation
<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 "2.11.6.1 Registering watch-expressions".

Remark The registered watch-expressions have their scope specification automatically given.

(b) Jump to memory

Selecting [Jump to Memory] from the 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, CS+ jumps to it directly.).

(c) Zoom in or out on a view

To zoom in and out of the Local Variables panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Local Variables panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(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	Sienna	Local variable values that have been changed due to program execution Note . Executing the program again will reset the highlighting color.
	Background color	LightYellow	
?	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 "2.11.5.2 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 "2.11.5.3 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.

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 in decimal (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel in 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.
	Adds the hexadecimal equivalent in bracket "( )" at the end of the value.
Encoding	Shows the following buttons that change the encoding in which string variables are displayed.
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.

[[File] menu (Local Variables 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 Variables 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 "2.11.5.2 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 displays the content of a registered watch-expression or changes the displayed value (see "2.11.6 Displaying and changing watch-expressions").

Up to four instances of this panel can be opened at a time. Each panel is discriminated by the name "Watch1," "Watch2," "Watch3" or "Watch4" in the title bar, allowing you to register, delete or move a watch-expression independently of the panel.

Watch-expressions can also be registered from the Editor panel, Disassemble panel, Memory panel, CPU Register panel, Local Variables panel or IOR panel, as well as from this panel.

If a panel that has watch-expressions registered in it is closed, the panel is hidden, but information on its registered watch-expressions is retained. (When the panel is reopened, it opens, with those watch-expressions registered in it as they were.)

When the value of any watch-expression changes after the program is executed, the display is updated. (During step execution, the display is updated successively as the program is stepped through.)

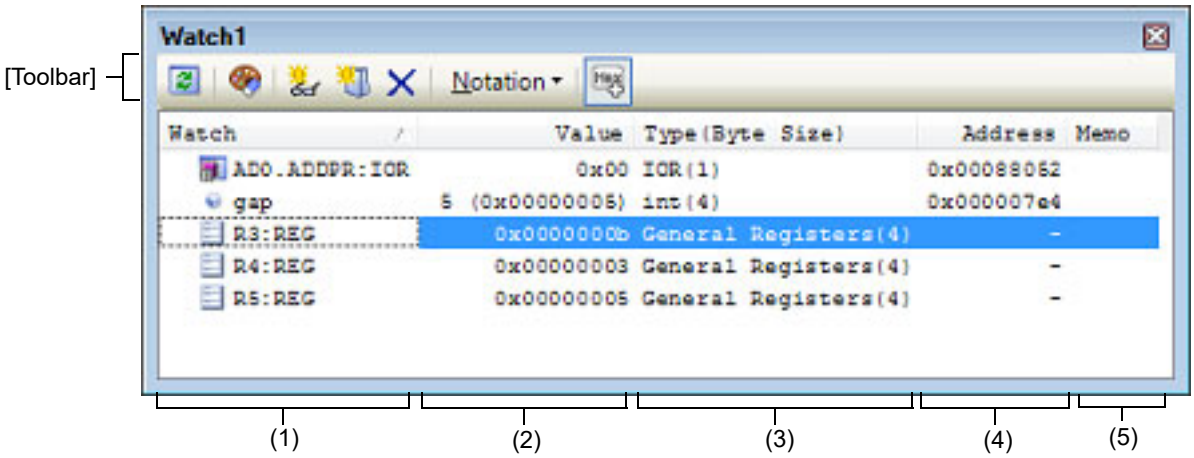
Also, if the Realtime display update function is enabled, the display of values can be updated in real time, even during program execution.

Note that this panel can only be opened when CS+ is connected with the debug tool.

Caution It is not possible to display or change the CPU register contents during program execution.

Remark By double-clicking a line delimiting each area on the panel, it is possible to change the relevant area to the smallest displayable width without omitting the content in it.

Figure A.13 Watch Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] Menu (Watch Panel-Only Items)]
- [[Edit] Menu (Watch Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Watch] from the [View] menu and then select [Watch*1-4*].

[Description of each area]

(1) [Watch] area

This area displays registered watch-expressions in list form.
By clicking the table title part of this area, it is possible to sort the watch-expressions in the list in alphabetical order.
Also, it is possible to create a category (folder) as desired and display the watch-expressions, as classified by category, in tree form (see "(a) Editing a tree").
The meaning of each displayed icon is as follows:

	Indicates a state that the watch-expressions belonging to this category are displayed. When you double-click the icon or click the "-" mark, the category is closed to hide the watch-expressions.
	Indicates a state that the watch-expressions belonging to this category are hidden. When you double-click the icon or click the "+" mark, the category is opened to display the watch-expressions.
	Indicates that the watch-expression is a variable. The watch-expression indicating an array, pointer type variable or a structure/union is marked with a "+" or "-" at the beginning, so that when the sign is clicked, the expression expands or collapses (see "(b) Expanding/collapsing the display").
	Indicates that the watch-expression is a function.
	Indicates that the watch-expression is an immediate.
	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 watch-expression having a lower level register (part of a register) is marked with a "+" or "-" at the beginning, so that when the sign is clicked, the expression expands or collapses (see "(b) Expanding/collapsing the display").

This area has the following features:

(a) Editing a tree

Watch-expressions can be classified by category (folder) for display in tree form.
To create a new category, move the caret to the position where you want to create and click the button in the toolbar or select [Create Category] on the context menu, and then enter any category name.
If the need arises to delete a category, select the category you want to delete, click the button in the toolbar or select [Delete] on the context menu.
Also, if you want to edit a category name you created, select the category name you want to edit and follow one of the following methods:

- Click the category name again and then edit it directly from the keyboard.
- Choose [Rename] 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.

After creating a category, drag-and-drop the registered watch-expressions into the category. The watch-expressions are displayed in tree form classified by category.
Similarly, the order in which categories or watch-expressions are displayed (one above or below the other) can be changed as desired by a drag-and-drop operation.

Caution 1. No other categories can be created within a category.

Caution 2. Up to 64 categories can be created in one Watch panel. (If an attempt is made to create more categories exceeding this limit, a message is displayed.)

Remark If categories or watch-expressions in a Watch panel are drag-and-dropped into another Watch panel (Watch1 to Watch4), the categories or watch-expressions are copied into the target Watch panel.

(b) Expanding/collapsing the display

The watch-expression indicating an array, pointer type variable, a structure/union/class or a register (only those that bear a name representing a part) is marked with a "+" at the beginning, so that when you click the sign, the next entry is expanded for display (when expanded, the sign changes to a "-").

Watch-expression	Contents when expanded
Array	All elements in an array. They can be displayed as a character string by selecting [Notation] and then [ASCII] on the context menu (up to 256 characters displayable). However, if not displayable because of encoding type, a "." or "?" is displayed.
Pointer type variable	A variable at the destination pointed to by a pointer.
Structure/union/class	All members of a structure/union/class.
Register	The name of a bit or bit string that comprises a register. Example: For PSW register IPL register, PM register, U register, I register, O register, S register, Z register, C register

(c) Registering a new watch-expression
There are following three methods of registering a new watch-expression.

<1> Registering a watch-expression from another panel
On another panel, perform one of the following operations on the subject that you want to register as a watch-expression:

- Select the subject character string and then drag-and-drop it directly into this area on any Watch panel (Watch1 to Watch4) (except for the Editor panel).
- Select the subject character string or move the caret to one of subject character strings (the subject being automatically selected), and then select [Register to Watch1] on the context menu.
- Choose [Copy] from the [Edit] menu to copy the subject character string and then choose [Paste] from the [Edit] menu in this area on any Watch panel (Watch1 to Watch4).

The table below shows the relationship between the panels for which this operation is possible and the subjects that are registrable as watch-expressions.

Table A.5 Relationship between Panels and Subjects Registrable as Watch-expressions

Panel name	Subjects registrable as watch-expressions
Editor panel	C/C++ variables[Note 1], CPU registers, I/O registers, and assembler symbols
Disassemble panel	C/C++ variables[Note 1], CPU registers, I/O registers, and assembler symbols
CPU Register panel	CPU registers [Note 2]
Local Variables panel	C/C++ variables[Note 1] (local variables)
IOR panel	I/O registers[Note 2]

Note 1. It represents C89, C99 and C++ variables.

Note 2. Scope specification is automatically added to the watch-expression.

<2> Direct registration on Watch panel
On any Watch panel (Watch1 to Watch4), click the button in the toolbar or select [Add New Watch] on the context menu. An entry box for new watch-expressions is displayed at the bottom of this area.
In the [Watch] area of the entry box, enter a watch-expression directly from the keyboard and then press the [Enter] key.
The form in which watch-expressions are entered this way is as follows:

Table A.6 Watch-expression Input Form

Watch-expression	Displayed value
C/C++ variable name Note 1	Value of a C/C++ variable
Watch-expression [*Watch expression*]	Element values of an array
Watch-expression. Member name Note 2	Member values of a structure/union/class
Watch-expression-> Member name Note 2	Member values of the structure/union/class pointed to by a pointer
Watch-expression.* *Cast expression*	Value of a pointer to member variable
Watch-expression->* *Cast expression*	Value of a pointer to member variable
* Watch-expression	Variable value of a pointer
& Watch-expression	Location address
(Type name) *Watch-expression*	Value cast to specified type
CPU register name	Value of a CPU register
I/O register name	Value of an I/O register
Label name Note 3, EQU symbol name Note 3, [immediate value]	Value of a label, value of an EQU symbol, value of an immediate address
Integer constant	Constant value of an integer
Floating constant	Constant value of a floating point
Character constant	Constant value of a character

Note 1. It represents C89, C99 and C++ variables.

Note 2. To specify a member variable of base class, specify scope before a member name. (Example: variable.BaseClass::member)

Note 3. If a label name or EQU symbol name contains a "$," enclose the name in braces "{ }." (Example: {$Label})
If there is an imaginary number, multiply it by the uppercase letter "I" (example: 1.0 + 2.0*I).
The letter "I" constitutes a keyword, so that when you specify "I" of a CPU register, add ":REG" (example: I:REG).

Also, scope can be specified for a watch-expression when it is registered. When a watch-expression is registered by specifying scope, it is handled as shown below.

Table A.7 Handling of a C/C++ Variable when Registered in Watch by Specifying Scope

Scope specification	Load module name Note 1	Source file name Note 1	Function name- Note 2	Subject to be searched
prog$file#func	prog	file	func	Static functions
prog$func	prog	Global	func	Global functions
file#func	Current	file	func	Static functions
func	Current	Current	func	All Note 2

Note 1. If a load module name or file name contains a space or one of the following symbols, enclose the name in double-quotes (" "). (Example: "c:\folder\prog.abs" $file.c#func)
$, #, (,), [,], &, ^, ~, %, +, - *, /, :, ?, ', |, \, <, >, !

Note 2. A search is made for static functions and global functions from the scope of the current PC value in that order. Static functions out of scope are not searched for.
To specify a function defined in a name space, do not write scope. (Example: Scope::func)

Also, if functions with the same name exist, write the type of parameter expressly. (Example: func(int, int))

Table A.8 Handling of a C Variable when Registered in Watch by Specifying Scope

Scope specification	Load module name[Note 1]	Source file name[Note 1]	Function name[Note 2]	Function name[Note 2]	Subject to be searched
prog$file#func#var	prog	file	func	var	Static variables inside a static function[Note 2,3]
prog$file#var	prog	file	Global	var	Static variables inside a file
prog$var	prog	Global	Global	var	Global variables
file#func#var	Current	file	func	var	Static variables inside a static function[Note 2,3]
file#var	Current	file	Global	var	Static variables inside a file
var	Current	Current	Current	var	All[Note 4]

Note 1. If a load module name or file name contains a space or one of the following symbols, enclose the name in double-quotes (" "). (Example: "c:\folder\prog.abs" $file.c#func#var)
$, #, (,), [,], &, ^, ~, %, +, - *, /, :, ?, ', |, \, <, >, !

Note 2. To specify a function and variable defined in a name space, be sure to write scope. (Example: Scope::func)
Also, if functions with the same name exist, write the type of parameter expressly. (Example: func(int, int))

Note 3. If the current PC value exists in a specified function, the local variables that are not declared as static also comprise the subject to be searched.

Note 4. A search is made for local variables, static variables inside a file and global variables from the scope of the current PC value in that order. The local variables and the static variables inside a file that are out of scope are not searched for.

Table A.9 Handling of a CPU Register when Registered in Watch by Specifying Scope

Scope specification	Register bank	CPU register name
R10:REG	(None)	R10

Table A.10 Handling of an I/O Register when Registered in Watch by Specifying Scope

Scope specification	I/O register name
AD0.ADCR:IOR	AD0.ADCR
AD0.ADCR	AD0.ADCR

Remark 1. A symbol name at the current caret position can be supplemented by pressing the [Ctrl] and [Space] keys together in this area. (See "2.21.2 Symbol name completion function.")

Remark 2. Immediates are handled as a numeric value. Also, operators can be used for immediates.

Remark 3. Arithmetic expressions using a symbol can be specified as watch-expressions.

Remark 4. If there exist a C/C++ variable, a CPU register and an I/O register with the same name that are registered without specifying scope, the symbols are resolved in the order given below, to display values.
C/C++ variable > CPU register > I/O register

Remark 5. If there exist a local variable and a global variable with the same name that are registered by only a symbol name without specifying scope, the symbols are resolved based on the scope of the current PC value, to display values.

Remark 6. If a watch-expression is registered from the IOR panel or CPU Register panel, the watch-expression will have scope specification automatically added to it.

Remark 7. If the letter "I" alone is specified as a watch-expression, it is interpreted as an imaginary key-word. To acquire the value of a register "I," add ":REG" after the register.

<3> Registering from another application
Select a character string representing a C/C++ variable, CPU register, I/O register or assembler symbol from an external editor or the like, and then perform one of the following operations:

- Drag-and-drop the subject character string into this area on any Watch panel (Watch1 to Watch4).
- Copy the subject character string to the clipboard and then select [Paste] on the [Edit] menu in this area on any Watch panel (Watch1 to Watch4).

Caution 1. Up to 128 watch-expressions can be registered in one Watch panel. (If an attempt is made to register more watch-expressions exceeding this limit, a message is displayed.)

Caution 2. In a block where a variable that is the subject of operation is not used, variable data may not always exist in the stack or register, due to optimization by a compiler. In this case, even when a variable that is the subject of operation is registered as a watch-expression, the displayed value remains marked with "?."

Remark 1. Watch-expressions registered on each Watch panel (Watch1 to Watch4) are managed separately from each other and are saved as user information on the project.

Remark 2. More than one watch-expression with the same name can be registered.

(d) Editing a watch-expression
To edit a registered watch-expression, double-click the subject watch-expression to put it into edit mode and then enter the content of editing directly from the keyboard. (Pressing the [Esc] key cancels the edit mode.)
When you've finished editing a watch-expression, press the [Enter] key to complete the editing.

(e) Deleting a watch-expression
Click the button in the toolbar or select [Delete] on the context menu to delete a selected watch-expression.

(f) Setting various events
By selecting [Access Break] or [Trace Output] on the context menu, it is possible to set various events in a selected watch-expression.
When an access-related break event is set, the icon of the watch-expression changes shape. (An event mark for break events is displayed additionally below the icon of the watch-expression.) If the event you set is a trace event, there are no changes to the mark of the watch-expression.
When an event is set, detail information on the set event is reflected in the Events panel.
However, events can be set only when the watch-expression for which you're going to set an event is a global variable, a static variable inside a function, a static variable inside a file, or an I/O register.
For details on how to set an event, see the sections listed below.

- "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“

(g) Jumping to a memory definition address
Selecting [Jump to Memory] on the context menu opens the Memory panel (Memory1), with the caret on it moved to the address where a selected watch-expression is defined. (If the Memory panel (Memory1) is already open, CS+ jumps to it directly.)
However, if multiple watch-expressions are selected at the same time or an I/O register or CPU register is selected, this operation has no effect.

(h) Zoom in or out on a view
To zoom in and out of the Watch panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Watch panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(2) [Value] area

This area displays or changes the value of a registered watch-expression.
Note that if the watch-expression is a function pointer, a function name is displayed.
The display notation (numeral representation) and encoding can be selected using a toolbar button or from the content menu. Also, it is possible to select a display form that always adds hexadecimal equivalents to the ordinary display.
Note that the default display form is automatically determined depending on the type of watch-expression, as follows:

Table A.11 Display Form of Watch-expressions (Default)

Type of watch-expression	Display form
char, signed char, unsigned char	Hexadecimal value added in "( )" following ASCII character
short, signed short, short int, signed short int, int, signed, signed int, long, signed long, long int, signed long int	Hexadecimal value added in "( )" following signed decimal value
unsigned short, unsigned short int, unsigned, unsigned int, unsigned long, unsigned long int	Hexadecimal value added in "( )" following unsigned decimal value
float, float _Complex, float _Imaginary	Float value (when 4 bytes in size) Note
double, long double, double _Complex, long double _Complex, double _Imaginary, long double _Imaginary	Double value (when 8 bytes in size) Note
Pointer to char, singed char, unsigned char	String Encoding: Shift_JIS
Pointer to other than char, singed char, unsigned char	Hexadecimal
Array of type char, signed char, unsigned char	String Encoding: Shift_JIS
bit, boolean, _boolean	Hexadecimal value added in "( )" following unsigned decimal value
Enumeration type	Hexadecimal value added in "( )" following enumerated constant
Label, immediate address, EQU symbol	Hexadecimal value added in "( )" following signed decimal value
Bit symbol	Hexadecimal value added in "( )" following unsigned decimal value
Other	Hexadecimal

Note The values of floating and complex types are rounded to an approximate value (nearest whole value) before being displayed.

The meaning of marks displayed as values of watch-expressions 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	Values of watch-expressions which have had their values changed by the user (Written into the target memory by hitting [Enter] key)
	Background color	Standard color	
0x0	Text color	DeepPink	Values of watch-expressions which have their display updated in real time (Realtime display update function)
	Background color	Standard color	
0x0	Text color	Sienna	Values of watch-expressions whose values have changed as a result of program execution Highlighting is reset by clicking the button in the toolbar or selecting [Reset Color] on the context menu.
	Background color	LightYellow	
?	Text color	Gray	When nonexistent variables are registered as watch-expressions, or when values of watch-expressions could not be acquired (e.g., variables gotten out of scope)
	Background color	Standard color	

Remark 1. The I/O registers that may get the microcontroller operated unintentionally by a read operation are protected against reads, so that they are not loaded with values. To load the contents of the I/O registers protected against reads, select [Force Read Value] on context menu.

Remark 2. Watch-expressions have their values acquired in the order they are registered.
Therefore, if two or more of the same I/O register are registered, their displayed values may not always the same, because their values are acquired with different timing.

Remark 3. If hexadecimal equivalents are added, the value in specified notation and its hexadecimal equivalent are read out separately.
Therefore, the value in specified notation and its hexadecimal equivalent may not always agree, because they are acquired with different timing.

This area has the following features:

(a) Realtime display update function
By using the realtime display update function, it is possible to display or change the value of a registered watch-expression, not only when the program is halted, but also when it is under execution.
For details about the realtime display update function, see "2.11.1.4 Displaying and changing memory contents during program execution."

(b) Changing the value of a watch-expression
To change the value of a watch-expression, select the value of the subject watch-expression and click on it again. Then enter a new value directly from the keyboard. (Pressing the [Esc] key cancels the edit mode.)
When you've finished editing the value of a watch-expression, press the [Enter] key or move the focus to other than the edit area. The value you've changed is written into the target memory.
For details on how to change the value of a watch-expression, see "2.11.6.6 Changing the contents of watch-expressions."

(c) Saving the values 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 entire content of this panel to a text file (*.txt) or CSV file (*.csv).
By selecting [Save Expanded Watch Data…] from the context menu, the Save As dialog box can be opened, and the selected contents of watch-expressions can be saved in a text file (*.txt) or CSV file (*.csv).
For details on how to save the values of watch-expressions, see "2.11.6.9 Saving the displayed contents of watch-expressions."

(3) [Type (Byte Size)] area
This area displays the type information of watch-expressions in the form shown below.

Watch-expression	Display form
Single CPU register	<*Type of CPU register*> (<*Size*[Note 1]>)

Watch-expression	Display form	
Single I/O register	*<Type of I/O register>* (*<Access attribute>* *<Access type><Size*[Note 1]*>*)	
	Access attribute	R: Read only W: Write only R/W: Read/write
	Access type	1: Accessible in 1 bit 8: Accessible in byte 16: Accessible in half word 32: Accessible in word
Not determinable	?	
Other than above	*<Type of watch-expression as determined by C compiler*[Note 2]*>* (*<Size*[Note 1]*>*)	

Note 1. The sizes of watch-expressions are expressed in byte units.
For bit IORs and C bit fields, however, the sizes are displayed in bit units, with the "bit" notation added at the end of the value.

Note 2. Watch-expressions are handled as having the type indicated here when they are compiled.
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
This area displays the address to which each watch-expression is mapped (always expressed in hexadecimal). However, if the watch-expression is a single CPU register or not determinable, it is marked with "-" or "?," respectively.

Remark If the watch-expression is an I/O register that is a bit register, a bit offset value is added to its displayed address, as shown below.

Example For a bit register mapped to bit 4 at the address "0xFF40"
Displayed content: 0xFF40.4

(5) [Memo] area
This area allows the user to enter a comment on a watch-expression or category, freely as desired.
The contents of comments entered in this area are held separately for each watch-expression and category, and are saved as user information on the project. Therefore, if one of these watch-expressions or categories is deleted, the content of its corresponding memo is also discarded.
Note that when an array, register or the like is expanded for display, you cannot enter a comment on any expanded element.
To edit a comment, double-click the item you want to edit. The selected item is placed in edit mode. (Pressing the [Esc] key cancels the edit mode.) A character string of up to 256 characters can be entered directly from the keyboard (new-line code ignored).
When you've finished editing a character string, press the [Enter] key or move the focus to other than the edit area to complete the character string editing.

[Toolbar]

	Reacquires all values of registered watch-expressions to update the display. However, the I/O registers protected against reads are not reloaded.
	Resets the highlighting for a selected watch-expression which indicates that its value has changed as a result of program execution. However, this is disabled during program execution.
	Registers a new watch-expression. Enter a watch-expression directly in the text box (see "(c) Registering a new watch-expression"). Note that up to 128 watch-expressions can be registered in one Watch panel.

	Adds a new category (folder). Enter a category name directly in the text box. Note that up to 64 categories can be created in one Watch panel. (No other categories can be created within a category.)
	Deletes a selected range of character string. If a watch-expression or category is selected, the item is deleted. However, this is disabled when an expanded item of a watch-expression is selected.
Notation	Shows the following buttons to change the display form of values.
AutoSelect	Displays the value of a selected watch-expression in per-variable predetermined notation (default) (see "Table A.11 Display Form of Watch-expressions (Default)").
Hexadecimal	Displays the value of a selected item in hexadecimal.
Signed Decimal	Displays the value of a selected item in signed decimal.
Unsigned Decimal	Displays the value of a selected item in unsigned decimal.
Octal	Displays the value of a selected item in octal.
Binary	Displays the value of a selected item in binary.
ASCII	Displays the value of a selected item in ASCII code.
Float	Displays the value of a selected item in Float. However, this is enabled only when a selected watch-expression consists of 4-byte data.
Double	Displays the value of a selected item in Double. However, this is enabled only when a selected watch-expression consists of 8-byte data.
	Adds a hexadecimal equivalent for the displayed value of a selected item at the end of it, with the equivalent enclosed in "( )." However, this does not apply when the value is displayed in hexadecimal.

[[File] Menu (Watch Panel-Only Items)]

The [File] menu used exclusively for the Watch panel is as follows. (The other items are shared.)
However, all of these items are disabled during program execution.

Save Watch Data	Saves the content of this panel to a text file (*.txt) or CSV file (*.csv) that has been saved previously (see "(c) Saving the values of watch-expressions"). Note that if this item is selected for the first time after startup, the same operation as [Save Watch Data As...] would have been selected is performed.
Save Watch 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 "(c) Saving the values of watch-expressions").

[[Edit] Menu (Watch Panel-Only Items)]

The [Edit] menu used exclusively for the Watch panel is as follows. (All other items are disabled.)

Cut	Cuts a selected range of character string and copies it to the clipboard. If a watch-expression or category is in selected state, the item is cut. However, this is disabled when an expanded item of a watch-expression is selected.
Copy	Copies a selected range of character string to the clipboard. If a watch-expression or category is in selected state, the item is copied. However, this is disabled when an expanded item of a watch-expression is selected.
Paste	Inserts the content of the clipboard into the caret position when text is in edit mode. If, when text is in other than edit mode, a watch-expression is copied to the clipboard, the copied watch-expression is registered at the caret position.

Delete	Deletes a selected range of character string. If a watch-expression or category is in selected state, the item is deleted. However, this is disabled when an expanded item of a watch-expression is selected.
Select All	Selects all character strings when text is in edit mode. If the text is in other than edit mode, all watch-expressions and categories are placed in selected state.
Rename	Edits the name of a selected watch-expression or 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	This item is usable when the selected watch-expression is a global variable, a static variable inside a function, a static variable inside a file, or an I/O register (multiple selection not accepted). Shows the following cascaded menu to set access-related break events (see "2.10.4.1 Set a break event (access-related) to a variable/I/O register").
Set Read Break to [Simulator]	Sets a read-access break event in a selected watch-expression.
Set Write Break to [Simulator]	Sets a write-access break event in a selected watch-expression.
Set R/W Break to [Simulator]	Sets a read/write-access break event in a selected watch-expression.
Set Read Combination Break to [E1] [E20] [EZ Emulator]	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] [EZ Emulator]	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] [EZ Emulator]	Sets a read/write-access break event in a selected watch-expression as the condition for a combination break event.

Trace Output	This item is usable when the selected watch-expression is a global variable, a static variable inside a function, a static variable inside a file, or an I/O register (multiple selection not accepted). Shows the following cascaded menu to set trace-related events. (see "2.13.3 Collecting an execution history in a section" and "2.13.4 Collecting an execution history only when conditions are met"). After an event is set, a watch-expression that is the subject of operation is marked with an (b) Event mark at the beginning.
Record Reading Value	Sets a point trace event that, when a selected watch-expression is accessed for read, records the accessed value in trace memory (see "(1) When an access to a variable or I/O register occurred").
Record Writing Value	Sets a point trace event that, when a selected watch-expression is accessed for write, records the accessed value in trace memory (see "(1) When an access to a variable or I/O register occurred").
Record R/W Value	Sets a point trace event that, when a selected watch-expression is accessed for read/write, records the accessed value in trace memory (see "(1) When an access to a variable or I/O register occurred").
Record Start R/W Value	Sets a trace event that, when a selected watch-expression is accessed for read/write, causes trace recording to start (see "2.13.3.1 Setting a trace start event and a trace end event").
Record End R/W Value	Sets a trace event that, when a selected watch-expression is accessed for read/write, causes trace recording to end (see "2.13.3.1 Setting a trace start event and a trace end event").
Trace	Opens the Trace panel to display acquired trace data.
Timer Settings[Note 1]	This item is enabled only when the selected watch-expression is a global variable, a static variable inside a function, a static variable inside a file, or an I/O register (plural selections not accepted). Displays the following cascaded menu to set timer-related events (see "2.14.3 Measuring execution time in a section"). After an event is set, an event mark is displayed at the beginning of the watch-expression concerned.
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 "(1) How to set a timer start event").
Set Timer <N> [E1] [E20] [EZ Emulator]	Specify a channel[Note 2] 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 "(2) How to set a timer end event").
Set Timer <N> [E1] [E20] [EZ Emulator]	Specify a channel[Note 2] in which a timer end event is set.
Periodic Updating	Shows the following cascaded menu to set realtime display updating (see "(a) Realtime display update function").
Periodic Updating Options	Opens the Property panel to set the realtime display update function generally.
Refresh	Reacquires all values of registered watch-expressions to update the display. However, the I/O registers protected against reads are not reloaded.
Force Read Value	Forcibly loads the values of I/O registers protected against reads once. However, this is disabled during program execution.

Add New Watch	Registers a new watch-expression. Enter a watch-expression directly in the text box (see "(c) Registering a new watch-expression"). Note that up to 128 watch-expressions can be registered in one Watch panel.
Create Category	Adds a new category (folder). Enter a category name directly in the text box. Note that up to 64 categories can be created in one Watch panel (no other categories can be created within a category).
Delete	Deletes a selected range of character string. If a watch-expression or category is in a selected state, the item is deleted. However, this is disabled when an expanded item of a watch-expression is selected.
Cut	Cuts a selected range of character string and moves it to the clipboard. If a watch-expression or category is in a selected state, the item is cut. However, this is disabled when an expanded item of a watch-expression is selected.
Copy	Copies a selected range of character string to the clipboard. If a watch-expression or category is in a selected state, the item is copied.
Paste	Inserts the content of the clipboard into the caret position when text is in edit mode. If, when text is in other than edit mode, a watch-expression is copied to the clipboard, the copied watch-expression is registered at the caret position. However, this is disabled when an expanded item of a watch-expression is selected.
Rename	Renames the selected watch-expression/category.
Import Watch Expression...	Opens the Open Watch Expression Data File dialog box to import watch-expressions (see "2.11.6.8 Exporting/importing watch-expressions").
Notation	Shows the following cascaded menu to specify a display form.
AutoSelect	Displays the value of a selected watch-expression in per-variable predetermined notation (default) (see "Table A.11 Display Form of Watch-expressions (Default)".)
Hexadecimal	Displays a selected item in hexadecimal.
Signed Decimal	Displays a selected item in signed decimal.
Unsigned Decimal	Displays a selected item in unsigned decimal.
Octal	Displays a selected item in octal.
Binary	Displays a selected item in binary.
ASCII	Displays a 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 "( )". However, this does not apply when the value is displayed in hexadecimal.
Float	Displays a selected item in Float. However, if a selected watch-expression is not 4-byte data or has type information, the selected item is displayed in predetermined notation (see "Table A.11 Display Form of Watch-expressions (Default)").
Double	Displays a selected item in Double. However, if a selected watch-expression is not 8-byte data or has type information, the selected item is displayed in predetermined notation (see "Table A.11 Display Form of Watch-expressions (Default)").
Decimal Notation for Array Index	Displays the indices of all arrays in decimal.
Hexadecimal Notation for Array Index	Displays the indices of all arrays in hexadecimal.

Encoding	Shows the following cascaded menu to specify character code.
ASCII	Displays a selected item in ASCII code.
Shift_JIS	Displays a selected item in Shift_JIS code (default).
EUC-JP	Displays a selected item in EUC-JP code.
UTF-8	Displays a selected item in UTF-8 code.
UTF-16	Displays a selected item in UTF-16 code.
Size Notation	Shows the following cascaded menu to specify a size.
1 Bytes	Displays a selected item as 8-bit data.
2 Bytes	Displays a selected item as 16-bit data.
4 Bytes	Displays a selected item as 32-bit data.
8 Bytes	Displays a selected item as 64-bit data.
Jump to Memory	Opens the Memory panel (Memory1), with the caret on it moved to the address at which a selected watch-expression is defined (see "(g) Jumping to a memory definition address").
Reset Color	Resets the highlighting for a selected watch-expression which indicates that its value has changed as a result of program execution. However, this is disabled during program execution.
Save Expanded Watch Data…	Opens the Save As dialog box to newly save the selected contents of watch-expressions to the specified text file (*.txt)/CSV file (*.csv) (see "(c) Saving the values of watch-expressions").

Note 1. [E1] [E20] [EZ Emulator]
The RX100 Series does not support timers.

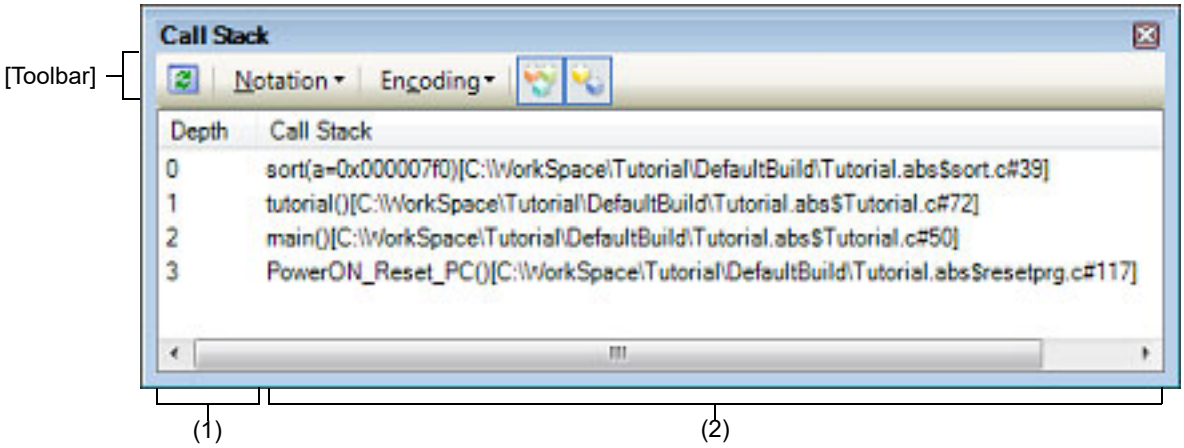
Note 2. The specifiable number of channels differs between the RX600/RX700 and RX200 Series, as shown below.
RX600, RX700 Series: 2 (32 bits * 2) or 1 (64 bits * 1)
RX200 Series: 1 (24 bits * 1)

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 CS+ is connected with the debug tool.

Caution This panel is left blank while the program is in execution.
The contents of each area on this panel are displayed at the time the program has stopped running.

Figure A.14 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 from the context menu, [Show Parameter] or [Show Module File Name], is selected, as shown below.

Condition	Display Format
- Show parameters - Show module file names	*<function>(<parameter>=<parameter value*[Note]*>, ...)[<module file name>$<file name>#<line number>]* (default)

Condition	Display Format
- Show parameters - Do not show module file names	*<function>(<parameter>=<parameter value*[Note]*>, ...)[<file Name>#<line number>]*
- Do not show parameters - Show module file names	*<function>()[<module file name>$<file name>#<line number>]*
- Do not show parameters - Do not show module file names	*<function>()[<file name>#<line number>]*

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 line
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 disassemble line indicating 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 the 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 "2.12.1.4 Saving the displayed contents of call stack information".

(c) Zoom in or out on a view
To zoom in and out of the Call Stack panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Call Stack panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

[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.

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 buttons that change the character code in which string variables 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.
	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 the 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 the 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 character code.
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 displays the trace data which has had an execution history of the program recorded in it (see "2.13 Collecting an Execution History").

Although the trace data is displayed, by default, in a disassembled text and source text mixed mode, it is possible to display either one of these text by selecting the desired Display mode.

The display position is automatically updated after program execution is halted, so that the latest trace data will be displayed.

Note that this panel can be opened only when CS+ is connected with the debug tool.

Caution [E20(JTAG) [RX600, RX700 Series]]
Part of the trace functions and Real-time RAM Monitor (RRM) functions can be used only on a mutually exclusive basis.

Remark By double-clicking a line delimiting each area on the panel, it is possible to change the relevant area to the smallest displayable width without omitting the content in it.

Figure A.15 Trace Panel [E1/E20/EZ Emulator [RX200, RX600, RX700 Series]]

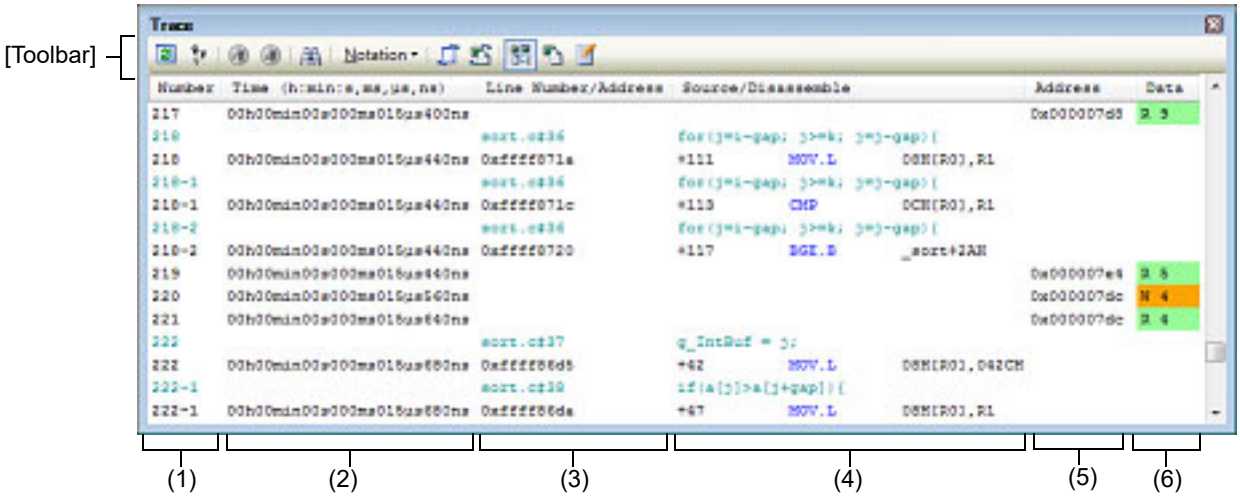


Figure A.16 Trace Panel [E1/E20/EZ Emulator [RX100 Series]]

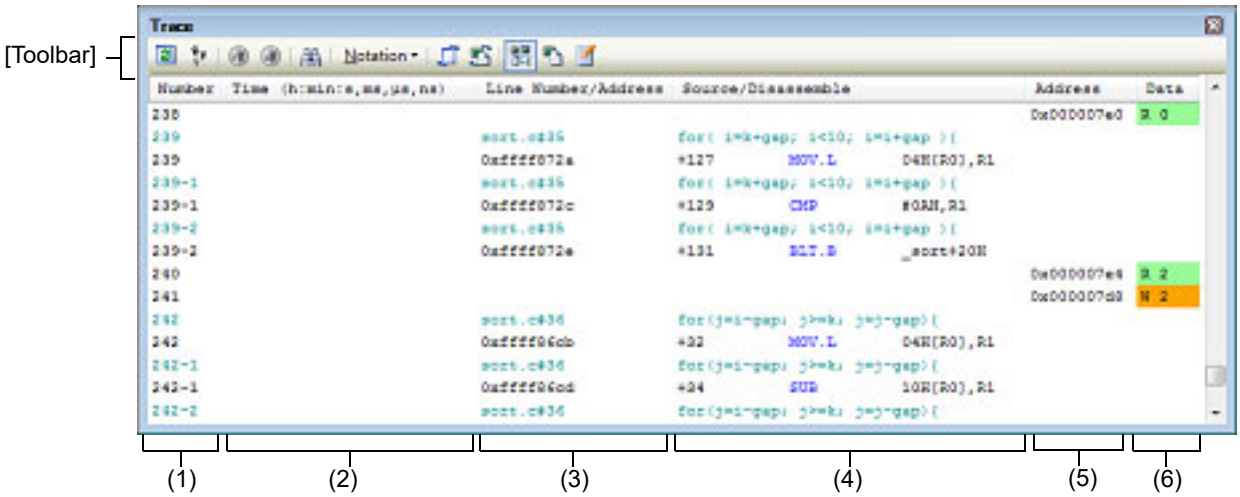
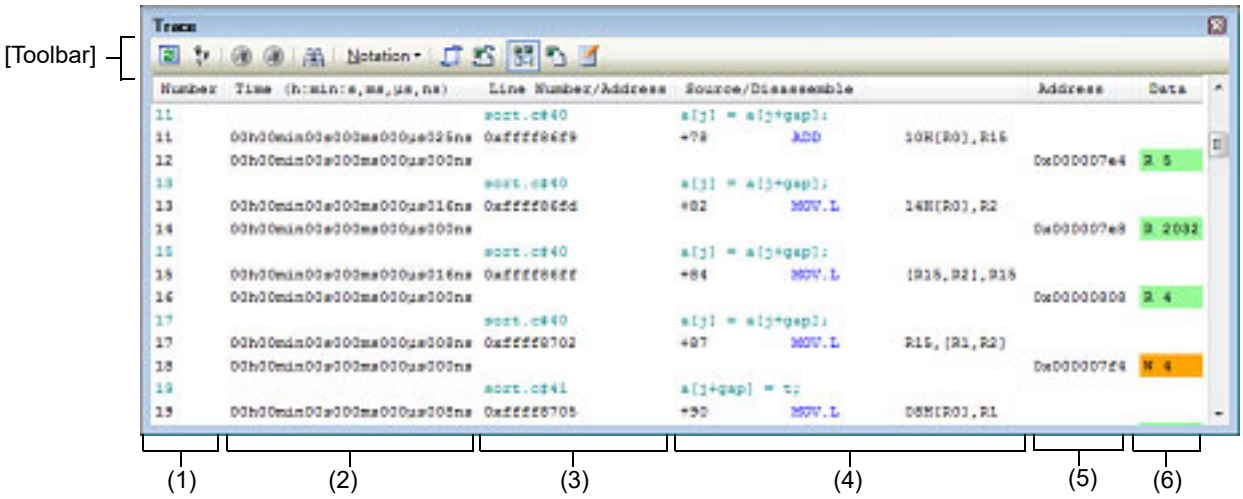


Figure A.17 Trace Panel [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] Menu (Trace Panel-Only Items)]
- [[Edit] Menu (Trace Panel-Only Items)]
- [Context menu]

[How to open]

- Choose [Trace] from the [View] menu.
- On the Editor panel or Disassemble panel, select [Trace Settings] and then [Show Trace Result] on the context menu.

[Description of each area]

(1) [Number] area
This area displays the trace numbers that correspond to trace frames.

(2) [Time (h:min:s,ms,µs,ns)] area
This area displays the time in "hour, minute, second, millisecond, microsecond and nanosecond" units from when the program started running till when the cause of instruction execution or memory access in any frame occurred.

Remark **[Simulator]**
Whether the time is displayed by an integral value or by a differential value depends on how the [Accumulate trace time] property in the [Trace] category on the Property panel's [Debug Tool Settings] tab is set.

Caution 1. [E1] [E20] [EZ Emulator]
To specify whether timestamp information is added to trace data, use the [Output timestamp] property in the [Trace] category on the Property panel's [Debug Tool Settings] tab. (See "2.13.1 Setting up a trace operation.") When an RX100-series microcontroller is in use, however, the [Output timestamp] property is fixed to [No] so timestamp information is not included in trace data.

Caution 2. [E1] [E20] [EZ Emulator]
The timestamp information displayed here is a time relative to the beginning cycle, and not a time from the start of program execution.

Caution 3. [E1] [E20] [EZ Emulator]
The timestamp information is calculated based on the set value of the [Trace clock count source[MHz]] property in the [Trace] category on the Property panel's [Debug Tool Settings] tab and a value counted with the inherent count source of each microcontroller used.

The counter width in bits of each microcontroller series is as follows:
RX600 series: 20 bits
RX200 series: 24 bits
Note that if the count value of the microcontroller overflows, an overflowed content is corrected before timestamp information is displayed.
However, if an overflow occurs twice or more between successive trace frames, correct timestamp information will not be displayed in the subsequent trace data.

Caution 4. [E1(Serial)/E20(Serial)/EZ Emulator [RX200 Series]]
A timer measurement counter is used for the Time Stamp in the trace panel. Therefore, if a timer measurement event is set on the Events panel, expected timestamp information will not be displayed.

Caution 5. The frequency for counting of timestamps in tracing varies (Trace clock count source[MHz]) with the microcontroller in use.

- [RX610, RX621, RX62N, RX62T, RX62G group microcontroller]
  (1) If EXTAL x 8 <= 100 MHz
  Trace clock count source[MHz] ＝ EXTAL × 8
  (2) If EXTAL x 8 > 100 MHz
  Trace clock count source[MHz] ＝ EXTAL × 4
- [RX71M, RX66x, RX65x, RX64M, RX630, RX631, RX63N, RX63T group microcontroller]
  (1) When EXTAL or PLL is used as the clock source for the system clock (ICLK)*
  (a) When the division ratio of 1:1 is set by SCKCR.ICK
  Trace clock count source[MHz] = Selected clock source /1
  (b) When a division ratio other than 1:1 is set by SCKCR.ICK
  Trace clock count source[MHz] = Selected clock source /1
  (2) When EXTAL or PLL is not used as the clock source for the system clock (ICLK)
  Trace clock count source[MHz] = Selected clock source /1
- [RX210, RX21A, RX220 group microcontroller]
  Trace clock count source[MHz] = ICLK (using the performance measurement counter)

*When the clock source for the system clock (ICLK) is changed while the user program is running, the count source for timestamps will also change. In such a case, simply multiplying the timestamp difference and the frequency does not make the absolute time.

(3) [Line/Address] area
This area displays the address of an assembly instruction or a line number in the source file.
The desired notation (numerical representation) or character string encoding can be selected by clicking the appropriate toolbar button or by selecting from a context menu.
The display form is as follows:

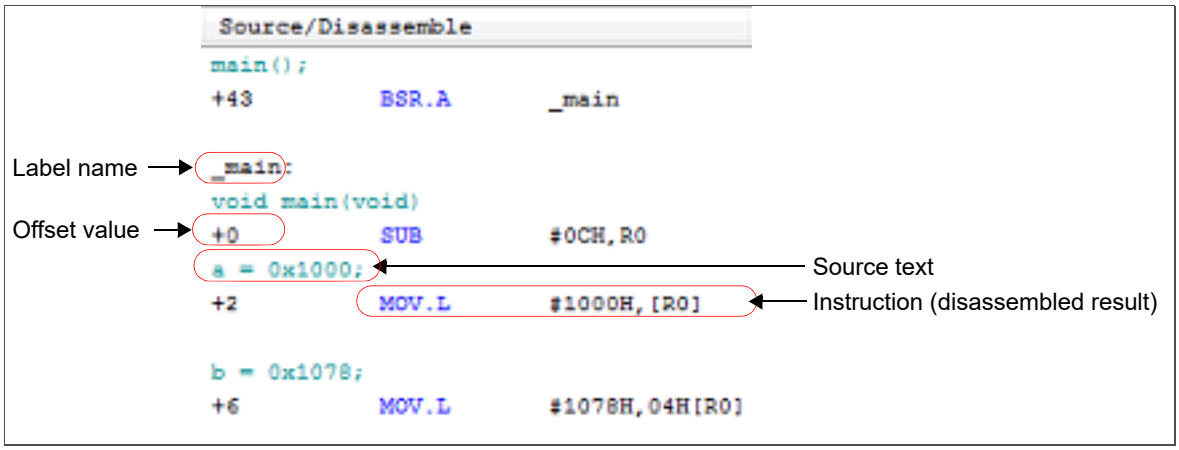
Type of displayed line	Display form
Instruction (disassemble)	*<Address>*
Source text	*<File name>#<Line number>*
Label	-
Point trace result	-

Remark Since the following execution histories are not displayed, the line numbers are not consecutive.

- CPU register access
- Invalid fetch

(4) [Source/Disassemble] area
This area displays collected trace data as shown below.
Note that the items displayed in this area differ with each display mode selected (see "(a) Display mode").

Figure A.18 Contents Displayed in the [Source/Disassemble] Area (Default)



Label name	If a label is defined at an address, the label name is displayed.
Offset value	If no label is defined at an address, an offset value from the nearest label is displayed.
Source text	If Mixed display mode or Source display mode is selected, the corresponding source text is displayed. However, if a location without debug information is executed, the text "No Debug Information" is displayed.
Instruction (disassembled result)	If Mixed display mode or Disassemble display mode is selected, the corresponding instruction (disassembled result) is displayed. The mnemonic is highlighted.

Note If collection of trace data failed, with some data dropped, the word "(LOST)" is displayed and the whole of the relevant line is displayed in an error color. (The error color depends on how the [General - Font and Color] category in the Option dialog box is set.)

This area has the following features:

(a) Display mode
One of the following three display modes can be selected by clicking the appropriate toolbar button or by selecting from a context menu.

Display mode	Displayed content
Mixed display mode	Instruction (disassemble), label name, source text (corresponding source line) and data access information are displayed (default).
Disassemble display mode	Instruction (disassemble), label name and data access information are displayed.
Source display mode	Source text (corresponding source line) is displayed. However, if a location without debug information is executed, the text "No Debug Information" is displayed.

(b) Jump to the source line/disassemble line
When [Jump to Source] on the context menu is selected, the Editor panel is opened, with the caret on it moved to a source line that corresponds to the line where the caret is currently positioned. (If the Editor panel is already open, CS+ jumps to it directly.)
Similarly, when [Jump to Disassemble] is selected, the Disassemble panel (Disasemmble1) is opened, with the caret on it moved to a fetch address on the line where the caret is currently positioned. (If the Disassemble panel (Disassemble1) is already open, CS+ jumps to it directly.)

(c) Linkage with other panels
By clicking the or button in the toolbar or selecting [Window Connecting] and then [Connect Source Window] or [Connect Disassemble Window] on the context menu, it is possible to have data corresponding to the address at the caret position on this panel displayed in synchronism with it on the Editor panel or Disassem-

ble panel. (The focus is not moved.)

(d) Popup display
When the mouse cursor is hovered on a line, the data in all areas (items) corresponding to the line are displayed one below another in a popup window.

(e) Saving of trace data
By choosing [Save Trace Data As...] from the [File] menu, it is possible to open the Data Save dialog box and then save the content of this panel to a text file (*.txt) or CSV file (*.csv).
For details on how to save trace data, see "2.13.9 Saving the displayed content of an execution history."

(f) Zoom in or out on a view
To zoom in and out of the Trace panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Trace panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(5) [Address] area
This area displays the subject address at which a memory access was made.
However, if an access was made to an I/O register, the I/O register name is displayed in place of an address. (If there are multiple accesses, the rest is displayed on the next line.)
The desired notation (numerical representation) can be selected by clicking the appropriate toolbar button or by selecting from a context menu.

(6) [Data] area
This area displays an accessed data value and the type of access.
However, this information is not displayed for accesses to the CPU register.
The desired notation (numerical representation) or character string encoding can be selected by clicking the appropriate toolbar button or by selecting from a context menu.
The display forms of data values and types of access are 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.)

Display example (default)			Type of memory access
R *Data value*	Text color	Standard color	Read access
	Background color	PaleGreen	
W *Data value*	Text color	Standard color	Write access
	Background color	Orange	
VECT*Data* [Simulator]	Text color	Standard color	Vector read access
	Background color	PaleGreen	

[Toolbar]

	Acquires latest information from the debug tool to update the display. However, this is disabled during program execution.
	Clears (initializes) the trace memory and also clears the display of this panel. However, this is disabled during program execution.
	Restarts the trace function that was halted during program execution. However, this is disabled when the program is halted or when tracer operation is underway.
	Temporarily stops the trace function during program execution. However, this is disabled when the program is halted or when the tracer is halted.
	Opens the Trace Search dialog box.
Notation	Shows the following buttons for changing the form in which values are displayed. However, this is disabled during program execution.
Hex	Displays values on this panel in hexadecimal (default).
Dec	Displays values on this panel in decimal.
Oct	Displays values on this panel in octal.
Bin	Displays values on this panel in binary.
	Scrolls the Editor panel in synchronism with a selected line.
	Scrolls the Disassemble panel in synchronism with a selected line.
	Changes the display mode to a Mixed display mode (default). However, this is disabled during program execution.
	Changes the display mode to a Disassemble display mode. However, this is disabled during program execution.
	Changes the display mode to a Source display mode. However, this is disabled during program execution.

[[File] Menu (Trace Panel-Only Items)]

The [File] menu used exclusively for the Trace panel is as follows. (The other items are shared.)
However, all of these items are disabled during program execution.

Save Trace Data	Saves the content of trace data to a text file (*.txt) or CSV file (*.csv) that has been saved previously (see "(e) Saving of trace data"). If this item is selected for the first time after startup, the same operation as [Save Trace Data As...] would have been selected is performed.
Save Trace Data As...	Opens the Data Save dialog box in order to save the content of trace data to a specified text file (*.txt) or CSV file (*.csv) (see "(e) Saving of trace data").

[[Edit] Menu (Trace Panel-Only Items)]

The [Edit] menu used exclusively for the Trace panel is as follows. (All other items are disabled.)
However, all of these items are disabled during program execution.

Copy	Copies the content of a selected line as a character string to the clipboard (multiple selection not accepted).
Find...	Opens the Trace Search dialog box.

[Context menu]

Clear Trace	Clears (initializes) the trace memory and also clears the display of this panel. However, this is disabled during program execution.
Start Trace	Restarts the trace function that was halted during program execution. However, this is disabled when the program is halted or when tracer operation is underway.
Stop Trace	Temporarily stops the trace function during program execution. However, this is disabled when the program is halted or when the tracer is halted.
Find...	Opens the Trace Search dialog box. However, this is disabled during program execution.
Copy	Copies the content of a selected line as a character string to the clipboard (multiple selection not accepted). However, this is disabled during program execution.
Mixed View	Changes the display mode to a Mixed display mode. However, this is disabled during program execution.
Disassemble View	Changes the display mode to a Disassemble display mode. However, this is disabled during program execution.
Source View	Changes the display mode to a Source display mode. However, this is disabled during program execution.
Notation	Shows the following cascaded menu to specify notation (numerical representation). However, this is disabled during program execution.
Hexadecimal	Displays values on this panel in hexadecimal (default).
Decimal	Displays values on this panel in decimal.
Octal	Displays values on this panel in octal.
Binary	Displays values on this panel in binary.
Window Connecting	Shows the following cascaded menu to link the display with other panels (see "(c) Linkage with other panels").
Connect Source Window	Scrolls the Editor panel in synchronism with a selected line.
Connect Disassemble Window	Scrolls the Disassemble panel in synchronism with a selected line.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1), with the caret on it moved to the fetch address on a selected line.
Jump to Source	Opens the Editor panel, with the caret on it moved to the source line corresponding on a selected line.
Jump to Memory	Opens the Memory panel, with the caret on it moved to the memory value corresponding to the line where the caret is positioned.

Events panel

This panel displays detailed information on the events that have been set on the Editor panel, Disassemble panel or Watch panel, as well as chooses to enable or disable event settings or deletes event settings (see "2.17 Event Management").

Note that this panel can be opened only when CS+ is connected with the debug tool.

Remark 1. For details about event settings, see "2.17.7 Points to note regarding event setting."

Remark 2. The events that have been set on the Function List panel or Variable List panel of the analysis tool are also managed on this panel.

Remark 3. By double-clicking a line delimiting each area on the panel, it is possible to change the relevant area to the smallest displayable width without omitting the content in it.

Figure A.19 Events Panel [E1] [E20] [Simulator]

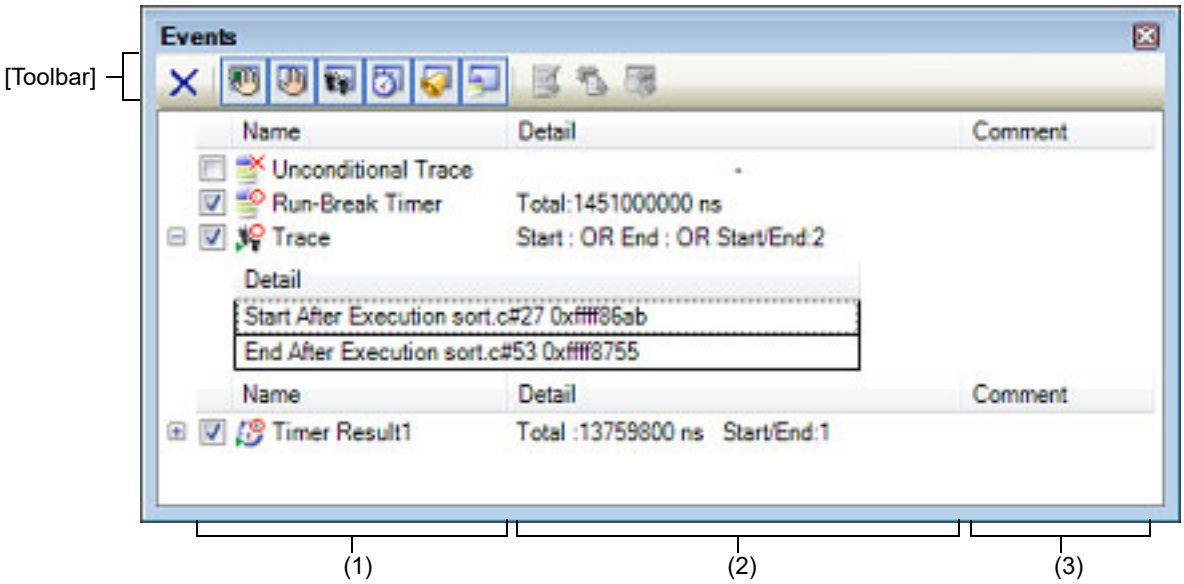
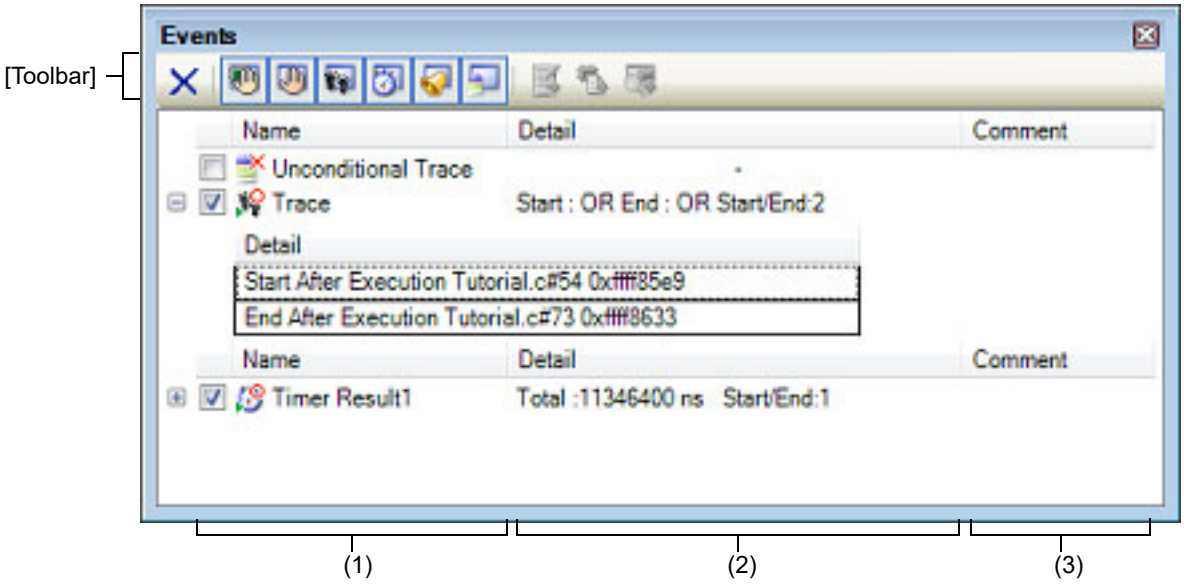


Figure A.20 Events Panel [EZ Emulator]



This section describes the following.

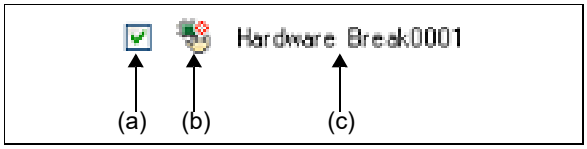
- [How to open]
- [Description of each area]
- [Toolbar]
- [[Edit] Menu (Event Panel-Only Items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Event].
- On the Editor panel or Disassemble panel, select [Timer Settings] and then [View Result of Timer] on 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] from the context menu.

[Description of each area]

(1) [Name] area
This area lists the currently set event names in the form shown below.



Remark By selecting the appropriate toolbar button, it is possible to limit the types of events displayed here (see [Toolbar]).

(a) Check box
Clicking the check box displays or changes the set state of an event.
Note that when the set state of an event is changed, the Event mark also changes accordingly.

	Enabled	When a specified condition becomes true, the event concerned with it occurs. The event can be disabled by unchecking the check box.
	Disabled	Even when a specified condition becomes true, the event concerned with it does not occur. The event can be enabled by checking the check box.
	Pending	A specified condition cannot be set in the program to be debugged. The check box cannot be manipulated.

Remark 1. The Run-Break timer event (not supported by the EZ Emulator) cannot be disabled or made pending.

Remark 2. The enabled or 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 or a trace end event is set, it is automatically disabled at the same time, in which case, a trace event (consisting of a trace start and a trace end event that are combined into one) is enabled.
Conversely, when the set trace event is disabled, the unconditional trace event is automatically re-enabled.

Caution [Simulator]
For the timer measurement event to be enabled, both timer start and end events should be specified.

(b) Event mark
The event mark indicates the type of an event and shows its currently set state.
The displayed event marks and their meanings are as follows:

Table A.12 Event Marks

Type of event	Enabled	Disabled	Pending	Remark
Hardware Break				-
Software Break [E1] [E20] [EZ Emulator]				-
Combination break **[E1] [E20] [EZ Emulator]**				-
Unconditional Trace			None	-
Run-Break Timer		None	None	-
Trace				Displayed on only the Events panel
Trace start				Displayed on only the Editor panel and Disassemble panel
Trace end				
Timer Result				Displayed on only the Events panel
Timer start				Displayed on only the Editor panel and Disassemble panel
Timer end				
Point Trace				-
Printf event				-
Interrupt event **[Simulator]**				-
Multiple settings of above events	Note 1	Note 2	Note 3	Displayed on only the Editor panel and Disassemble panel

Note 1. This applies when there is at least one enabled event among multiple events.

Note 2. This applies when there is no enabled event but at least one disabled event among multiple events.

Note 3. This applies when all of multiple events are made pending.

(c) Event name

The type of an event and its ID number are displayed as an event name.
The ID number is automatically assigned beginning with 0001 for each event type. (Even when an event that was once set is deleted, the other events are not reassigned ID numbers.)
The displayed event types are as follows:

Table A.13 Event Types

Event Type	Description
Hardware Break (Break[Note1])	This event is such that the debug tool successively checks break condition during program execution and, when the condition is met, causes the program to break. -> 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] [EZ Emulator]	This event is such that instruction code at an address where a break is to occur is rewritten with a break instruction, so that when the instruction is executed, the program is made to break. -> See "2.10.2 Stop the program at the arbitrary position (breakpoint)"

Event Type	Description
Combination break **[E1] [E20] [EZ Emulator]**	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] [EZ Emulator]"
Unconditional Trace	Trace data is automatically collected at the same time the program starts running, and collection of trace data is halted simultaneously when the program stops. Since this event is a built-in event Note2, it cannot be deleted. (This event is Enabled by default.) -> See "2.13.2 Collecting an execution history up to a halt"
Run-Break Timer	A measurement of the program's execution time automatically begins at the same time the program starts running, and the measurement of execution time is halted simultaneously when the program stops. Since this event is a built-in event Note2, it cannot be deleted. (This event is Enabled by default.) -> See "2.14.2 Measuring execution time from start to stop"
Trace	This event is such that when the condition set by a trace start and a trace end event is met, collection of trace data begins and then ends (displayed when either a trace start or a trace end event is set). -> See "2.13.3 Collecting an execution history in a section"
Timer Result Note3	This event is such that when the condition set by a timer start and a timer end event is met, a measurement of the program's execution time begins and then ends (displayed when either a trace start event or a trace end event is set). -> See "2.14.3 Measuring execution time in a section"
Point Trace	This event is such that when a specified variable or I/O register is accessed by program execution, the accessed information is recorded in trace memory. -> See "2.13.4 Collecting an execution history only when conditions are met"
Printf	This event is such that program execution is momentarily halted at any location and then a printf command is executed by a software process (i.e., an 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]"

Note 1. A breakpoint that is set by clicking the mouse once is displayed as "Break" (see "2.10.2.2 Set a breakpoint").

Note 2. This is an event that is set by default in the debug tool.

Note 3. **[E1] [E20] [EZ Emulator]**
Timer measurement sections are assigned channel numbers for each event type as their identification numbers (example: Timer Result 1).
The number of sections in which timer measurements are possible differs with each microcontroller used. Note the RX100 Series does not support timers.
- RX600, RX700 Series: Two sections
- RX200 Series: One section

[Simulator]
Only one section is available for timer measurement.

Remark In addition to the above event types, following event types may be displayed if events (breakpoints or break events) have been set on the Function List panel or Variable List panel of the analysis tool.

- For breakpoints for a function: "Break at start of function"
- For break events for a variable: "Access break to variable"

(d) Zoom in or out on a view
To zoom in and out of the Event panel view, change the zoom ratio by using the drop-down list on the toolbar of the Events panel while the focus is placed in the Event panel.
You can also change the zoom ratio 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%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(2) [Detail] area
This area displays detailed information on each event.
The content of displayed information differs with event type.
The following shows how to read detailed information for each event type.

Table A.14 Detailed Information by Event Type

Type of event	Displayed content Note1	
Hardware Break (Generation condition: Execution-related)	Display form 1	*<Generation condition> <File name#Line number> <Address>*
	Display example	Before Execution main.c#39 0x100
		Before Execution - 0x300
	Display form 2	*<Generation condition> <Symbol + Offset> <Address>*
	Display example	Before Execution funcA + 0x10 0x100
		Before Execution - 0x300
Hardware Break (Generation condition: Access-related)	Display form 1	*<Generation condition> <File name#Variable name> <Address(range)> <Comparison condition> <Comparison value>*
	Display example	Read main.c#variable1 0x100 - 0x101 == 0x5
		Write sub.c#variable2 0x200 - 0x200 == 0x7
		Read/Write sub2.c#variable3 0x300 - 0x303 == 0x8
	Display form 2	*<Generation condition> <File name#Function name#Variable name> <Address(range)> <Comparison condition> <Comparison value>*
	Display example	Read main.c#func1#variable1 0x100 - 0x101 == 0x10
	Display form 3	*<Generation condition> <Variable name> <Address(range)> <Comparison condition> <Comparison value>*
	Display example	Write variable1 0x100 - 0x101 == 0x10

Type of event	Displayed content [Note1]	
Software Break [E1] [E20] [EZ Emulator]	Display form1	*<Generation condition> <File name#Line number> <Address>*
	Display example	Before Execution main.c#40 0x102
		Before Execution sub.c#101 0x204
	Display form 2	*<Generation condition> <Symbol + Offset> <Address>*
	Display example	Before Execution funcA + 0x12 0x102
Combination break (Generation condition: Execution-related, Access-related) [E1] [E20] [EZ Emulator]	Display form	*<Combination condition>* *<Detailed information on combination break>*
	Display 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	Display form	-
	Display example	-
Run-Break Timer	Display form	Total: *<Total execution time>*
	Display example	Total: 1000ms
		Total: OVERFLOW
	Display form 2	Total:*<Total execution time>* Execution Cycle Count:*<Total number of cycles executed>* Execution Instruction Count:*<total number of instructions executed>* **[Simulator]**
	Display example	Total: 3300nsExecution Cycle Count:330 Execution Instruction Count:200
Trace (Generation condition: Execution-related, Access-related) [E1] [E20] [EZ Emulator]	Display form	Start: *<Combination condition for trace start>* End: *<Combination condition for trace end>* Start/End: *<Total number of trace start/trace end event>*[Note2] *<Start/end> <Detailed information on trace start/trace end>*
	Display example	Start: OR End: OR 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

Type of event	Displayed content Note1	
Trace (Generation condition: Execution-related, Access-related) [Simulator]	Display form	Start: OR End: OR Start/End: <*Total number of trace start/ trace end events*>Note2 <*Start/end*><*Detailed information on trace start/trace end*>
	Display example	Start: OR End: OR Start/End: 6 - Start Before Execution main.c#100 0x300 - Start Before Execution funcA + 0x100 0x300 - Start Write variable1 0x100-0x101==0x10 - End Before Execution main.c#200 0x100 - End Before Execution funcA + 0x10 0x100 - End Read main.c#variable1 0x100-0x101==0x5
Timer Result (Generation condition: Execution-related, Access-related) [E1] [E20] [EZ Emulator]	Display form	Total:<*Total execution time* > Start/End: <*Total number of timer starts/ timer ends*>Note 2 - <*Total execution time*> <*Pass count*> - <*Start/end*> <*Detailed information on timer start/timer end*>
	Display example	Total: 10ms Start/End: 6 - Total: 10ms Pass Count: 5 - Start After Execution main.c#100 0x300 - Start After Execution funcA + 0x30 0x100 - Start Write variable1 0x100-0x101==0x10 - End After Execution main.c#100 0x300 - End After Execution funcA + 0x50 0x100 - End Read main.c#variable1 0x100-0x101==0x5
Timer Result (Generation condition: Execution-related, Access-related) [Simulator]	Display form	Total:<*Total execution time* > Start/End: <*Total number of timer starts/ timer ends*>Note 2 - <*Total execution time*> <*Pass count*> - <*Start/end*> <*Detailed information on timer start/timer end*>
	Display example	Total: 10ms Start/End: 2 - Total: 10ms Pass Count: 5 - Start Write variable1 0x100-0x101==0x10 - End Before Execution main.c#100 0x300
Point Trace (Generation condition: Access-related)	Display form 1	<*Generation condition*> <*Variable name*> <*Address of variable*>
	Display example	Read variable1 0x100
	Display form 2	<*Generation condition*> <*File name#Variable name*> <*Address of variable*>
	Display example	Write sub.c#variable2 0x200
	Format3	<*Generation condition*> <*File name*#Function name#*Variable name*> <*Address of variable*>
	Display example	Read/Write sub.c#func1#variable3 0x300

Type of event	Displayed content Note1	
Printf event (Action event)	Display form	*<Generation condition>* *<File name#Line number>* *<Address>* *<Print event setting>*
	Display 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]	Display form	*<Generation condition>* *<File name # line number>* *<Address>* interrupt vector: *<Interrupt vector>* Priority level: *<Interrupt priority>*
	Display example	Before execution main.c#39 0x100 Interrupt vector: 1c Priority level: 7

Note 1. Following are the details on the display format.

<Generation condition>	One of the following conditions is displayed: [E1] [E20] Execution-related: Before Execution, After Execution Access-related: Read, Write, Read/Write [Simulator] Execution-related: Before Execution, Access-related: Read, Write, Read/Write
<File name # Line number>	A source file name and a line number in the source file are displayed. The display form is the same as for the scope specification expression of a watch-expression. If there are multiple load module files that are downloaded, *<Load module name$File name#Line number>* is displayed. For events set on the Disassemble panel, however, a *<Line number>* is displayed in the form *<Symbol + Offset>*, if the following applies: - Line information exists and a specified event set position is not at the beginning of line information. - No line information exists, but symbol information exists. Also, a *line number* is displayed with "-," if the following applies: - No line information exists, and no symbol information exists.
<Variable name>	A variable name in a source file is displayed. The display form is the same as for the scope specification expression of a watch-expression.
<Comparison condition>	A comparison condition (==) is displayed. This is not displayed if no comparison value is specified.
<Comparison value>	A comparison value is displayed. This is not displayed if no comparison value is specified.
<Address>	The beginning and the end address of a specified variable in a memory area are displayed (always in hexadecimal).
<Combination condition>	One of the following conditions is displayed: OR, AND, Sequential
<Detailed information on combination break>	Detailed information on combination break event is displayed.
<Total execution time>	The total execution time measured by a timer is displayed. The time is expressed in one of ns/μs,/ms/s/min units. (However, if expressed in "min," a "s" digit is displayed at the same time.) If the timer has overflowed (see "2.14.4 Range of measurable time") or the value is invalid, "OVERFLOW" is displayed.
<Combination condition for trace start>	One of the following conditions is displayed: OR, AND, Sequential

<Combination condition for trace end>	An OR condition is displayed.
<Total number of trace start/trace end event>	The total number of trace start and trace end events is displayed.
<Start/end>	Displays whether the content of the detailed information is about start event or end event.
<Detailed information on trace start/trace end>	Detailed information on trace start and trace end events is displayed.
<Pass count>	The pass count of a timer is displayed. If the timer has overflowed (see "2.14.4 Range of measurable time") or the value is invalid, "OVERFLOW" is displayed.
<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.
<Print event setting>	The *<Output character string>*: *<Variable expression>* specified in the Action Events dialog box is displayed.
<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].

Note 2. When this line is clicked, detailed information on the lines below that is displayed.

(3) [Comment] area
This area permits the user to enter freely a comment on any event that is set.
To enter a comment, select the event for which you want to enter a comment and click in this area or select [Edit Comment] on the context menu, and then enter any text directly from the keyboard. (Pressing the [Esc] key cancels the editing mode.)
When you've finished editing a comment, press the [Enter] key or move the focus to other than the edit area to complete your editing.
Note that a comment can be entered for up to 256 characters each, and the comment is saved as the one set by the current user.

[Toolbar]

	Deletes a selected event or an event condition. However, the built-in events (unconditional trace and Run-Break timer events) cannot be deleted.
	Displays hardware break-related events (default).
[E1] [E20] [EZ Emulator]	Displays software break-related events (default).
	Displays trace-related events (default).
	Displays timer-related events (default).
	Displays action events (Printf event or interrupt event) (default).
	Displays built-in events (unconditional trace and Run-Break timer events) (default).

	Opens the Editor panel, with the caret on it moved to the source line corresponding to the address where a selected event[Note] is set.
	Opens the Disassemble panel (Disassemble1), with the caret on it moved to the disassembled result corresponding to the address where a selected event[Note] is set.
	Opens the Memory panel (Memory1), with the caret on it moved to the memory value corresponding to the address where a selected event[Note] is set.

Note This does not include trace events, timer measurement events, and built-in events (unconditional trace and Run-Break timer events).

[[Edit] Menu (Event Panel-Only Items)]

The [Edit] menu used exclusively for the Events panel is as follows. (All other items are disabled.)

Delete	Deletes a selected event. However, the built-in events (unconditional trace and Run-Break timer events) cannot be deleted.
Select All	Selects all of the events displayed on 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]

Enable Event	Enables a selected event. However, this is ignored if the selected event is already enabled.
Disable Event	Disables a selected event. However, this is ignored if the selected event is already disabled.
Delete	Deletes a selected event or an event condition. However, the built-in events (unconditional trace and Run-Break timer events) cannot be deleted.
Select All	Selects all of the currently displayed events.
View Select	Shows the following cascaded menu to limit the event types to be displayed. By default, all items are selected.
Hardware Break	Displays hardware break-related events.
Software Break	Displays software break-related events.
Timer Event	Displays timer-related events.
Trace Event	Displays trace-related events.
Action Event	Displays action events (Printf event).
Built-in Event	Displays built-in events (unconditional trace and Run-Break timer events).
Timer Settings	Shows the following cascaded menu to make timer-related settings. However, this is enabled only when a timer-related event is selected.

Init Timer	Initializes the timer to be used in a selected event (not including the Run-Break timer event).
Nanosecond	Displays the result of a selected event measured by a timer in nanosecond (ns) units.
Microsecond	Displays the result of a selected event measured by a timer in microsecond (μs) units.
Millisecond	Displays the result of a selected event measured by a timer in millisecond (ms) units.
Second	Displays the result of a selected event measured by a timer in second (s) units.
Minute	Displays the result of a selected event measured by a timer in minute (min) units.
Jump to Memory	Opens the Memory panel (Memory1), with the caret on it moved to the memory value corresponding to the address where a selected event[Note1] is set.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1), with the caret on it moved to the disassembled result corresponding to the address where a selected event[Note1] is set.
Jump to Source	Opens the Editor panel, with the caret on it moved to the source line corresponding to the address where a selected event[Note1]is 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] [EZ Emulator] is opened. If a Printf event is selected, the Action Events dialog box is opened.
Edit Comment	Places a comment on a selected event into editing mode. If a comment for the event already exists, the whole of its character string is selected.

Note 1. This does not include trace events, timer measurement events, and built-in events (unconditional trace and Run-Break timer events).

Note 2. This includes the following events:

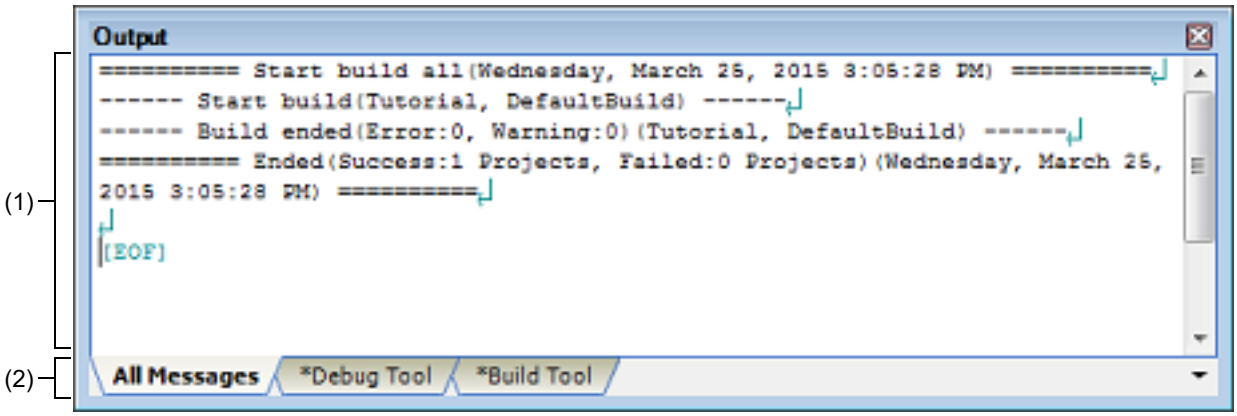
- Hardware break/execution-related events, access-related events, and timer measurement events **[E1] [E20]**

Output panel

This facility displays the messages output by various components supplied with CS+ (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.21 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	Displayed when unable to execute for a fatal error or operational mistake.
		Background color	WhiteSmoke	

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 for Message] from the 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.)

(d) Zooming in or out on a view
To zoom in and out of the Output panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Output panel.
You can also change the zoom ratio 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. 25%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

(2) Tab selection 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 CS+ (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 CS+ (e.g., design, build and analysis tools, including the debug tool).
Search 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.)
Note that 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. This is disabled during build execution.
Save Output-*tab 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.

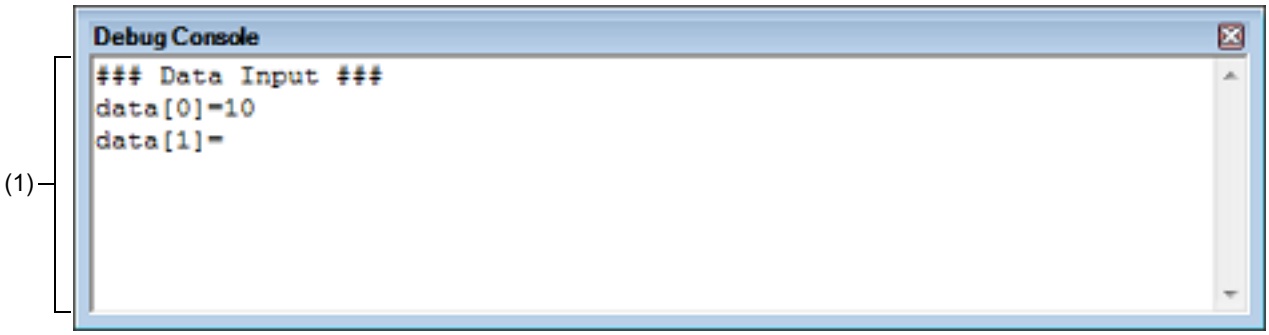
[Context menu]

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 and jumps 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 CS+ is connected with the debug tool.

Figure A.22 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) I/O 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 I/O 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 "2.20 Using the Debug Console").

Remark **[Simulator]**
For details about the I/O functions provided by the simulator, see "B. I/O FUNCTIONS."

(a) Zoom in or out on a view
To zoom in and out of the Debug Console panel view, change the zoom ratio by using the drop-down list on the toolbar of the Main window while the focus is placed in the Debug Console panel.
You can also change the zoom ratio 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. 25%).

If the panel is closed after the zoom ratio is changed, the changed zoom ratio is retained (next time, the panel will open at the changed zoom ratio).

[[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 echo back (a toggle switch). If local echoback is disabled, the input data is not output to this panel.

Memory Mapping dialog box

This dialog box sets memory mappings separately for each memory type.

Figure A.23 Memory Mapping Dialog Box [E1] [E20] [EZ Emulator]

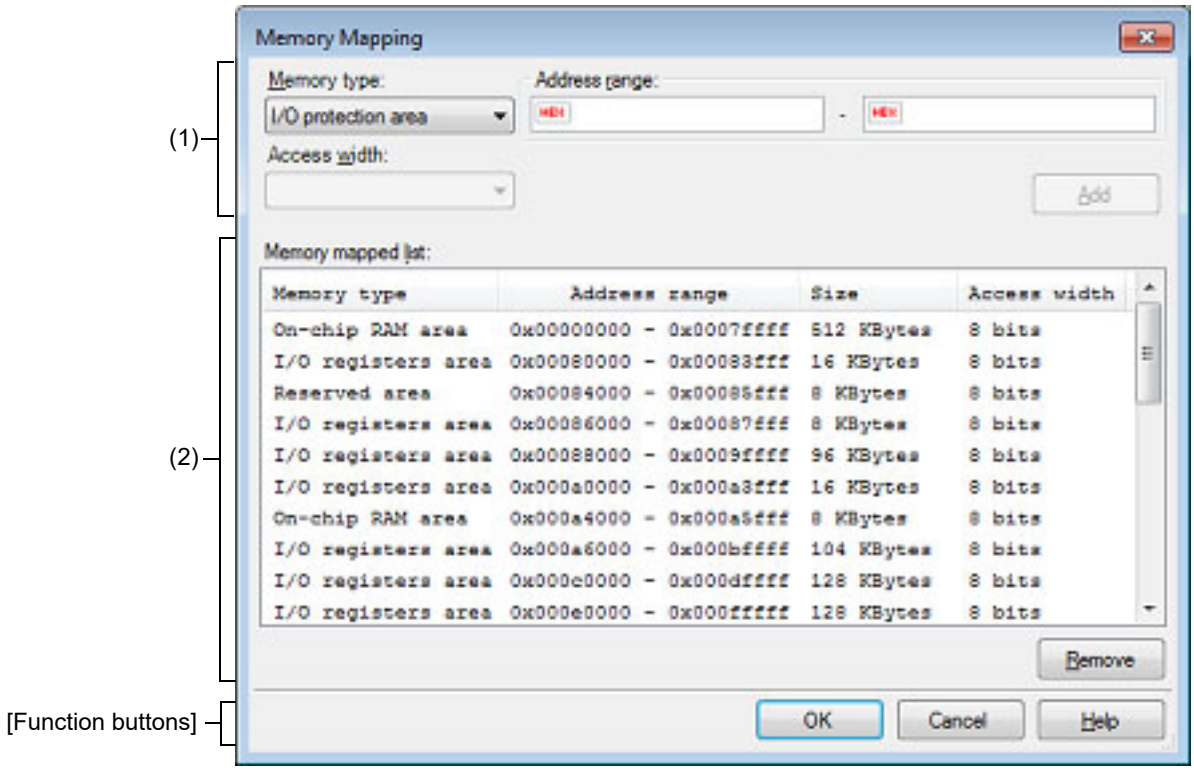
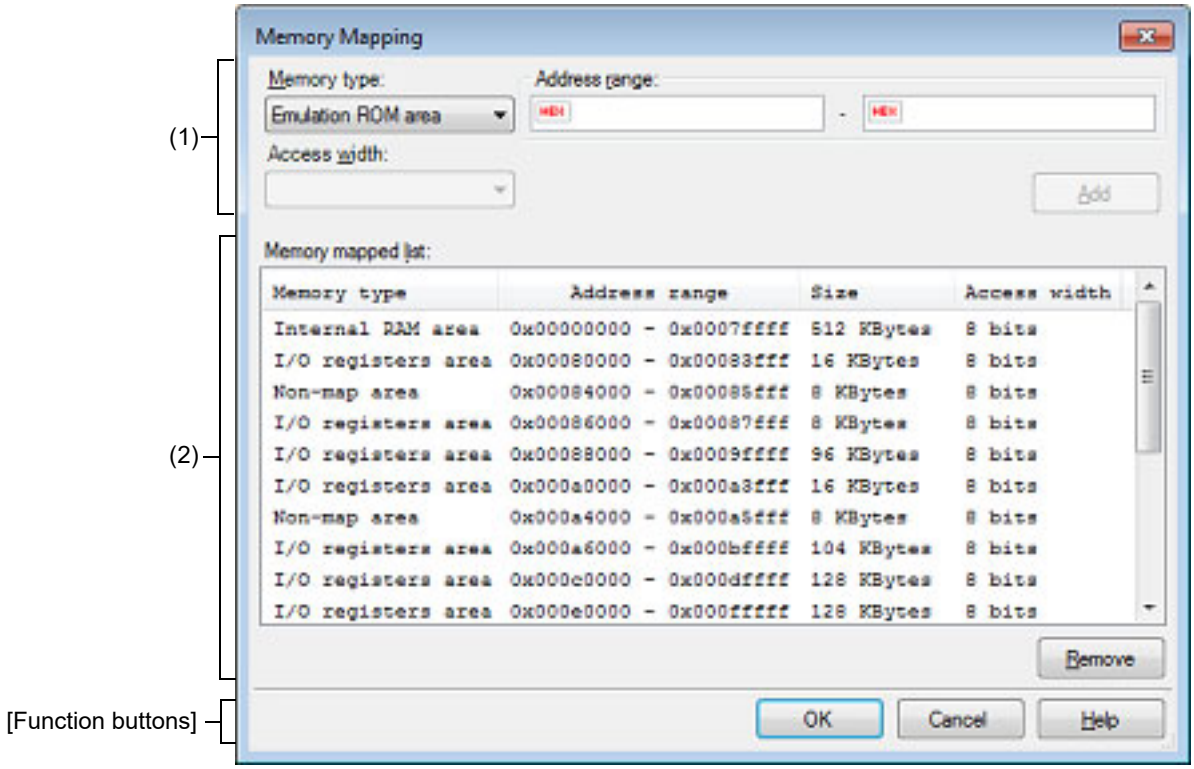


Figure A.24 Memory Mapping Dialog Box [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Click the [...] button that is displayed when you select the [Memory mappings] property in the [Memory] category on the Property panel's [Debug Tool Settings] tab.

Caution 1. [Simulator]
This dialog box cannot be opened during program execution.

Caution 2. [E1] [E20] [EZ Emulator]
This dialog box cannot be opened after a debug tool is connected.

[Description of each area]

(1) Additional memory mapping specification area
Specify information on a new memory mapping to be added.

(a) [Memory type]
Select the memory type of a memory mapping to be added from the drop-down list shown below. (The items selected by default depend on the debug tool.)

Emulation ROM area [Simulator]	Adds an emulation ROM area. Uses simulator alternative ROM.
Emulation RAM area [Simulator]	Adds an emulation RAM area. Uses simulator alternative RAM.
I/O protection area [E1] [E20] [EZ Emulator]	Adds an I/O protection area. Specify an address range in the external area that is not to be read by the debugger.

Caution 1. [Simulator]
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.

Caution 2. [Simulator]
Both read and write accesses to the memory mapping that has been added take one cycle each.

Caution 3. [E1] [E20] [EZ Emulator]
When the [OK] button is clicked on, each I/O protection area that has been added is adjusted to the access width of the external area that includes that I/O protection area.

(b) [Address range]
Specify the beginning and the end addresses of a memory mapping to be added. Enter a hexadecimal value directly in the respective text boxes.
When an emulator is in use, I/O protection areas can be allocated to space that overlaps with external areas.
Each I/O protection area cannot be allocated to two or more external areas.
When the simulator is in use, 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 mappings. The added memory mapping is displayed in the [Memory mapped list] area. Note that the changed content is not set until the [OK] button is pressed.

(2) [Memory mapped list] area

(a) List display

Displays memory mappings added in the Additional memory mapping specification area and information on the microcontroller's internal memory mappings. This area cannot be edited.

Memory type	Following memory types are displayed: - Internal ROM area - Internal RAM area - I/O register area - External area (CS7/CS6/.../CS0) [E1] [E20] [EZ Emulator] - Other memory area [E1] [E20] [EZ Emulator] - Reserved area [E1] [E20] [EZ Emulator] - I/O protection area [E1] [E20] [EZ Emulator] - Emulation ROM area [Simulator] - Emulation RAM area [Simulator] - Non-map area [Simulator]
Address range	Displays an address range <*Start address*> - <*End address*>. The addresses are always expressed in hexadecimal, with "0x" added.
Size	Displays a size in decimal (in byte or Kbyte units).
Access width	Displays an access width (in bit units)[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 a memory mapping selected in this area. When an emulator is in use, only I/O protection areas can be deleted. When the simulator is in use, the memory area that can be deleted is either an Emulation ROM area or an Emulation RAM area. (The microcontroller's internal memory mappings cannot be deleted.)

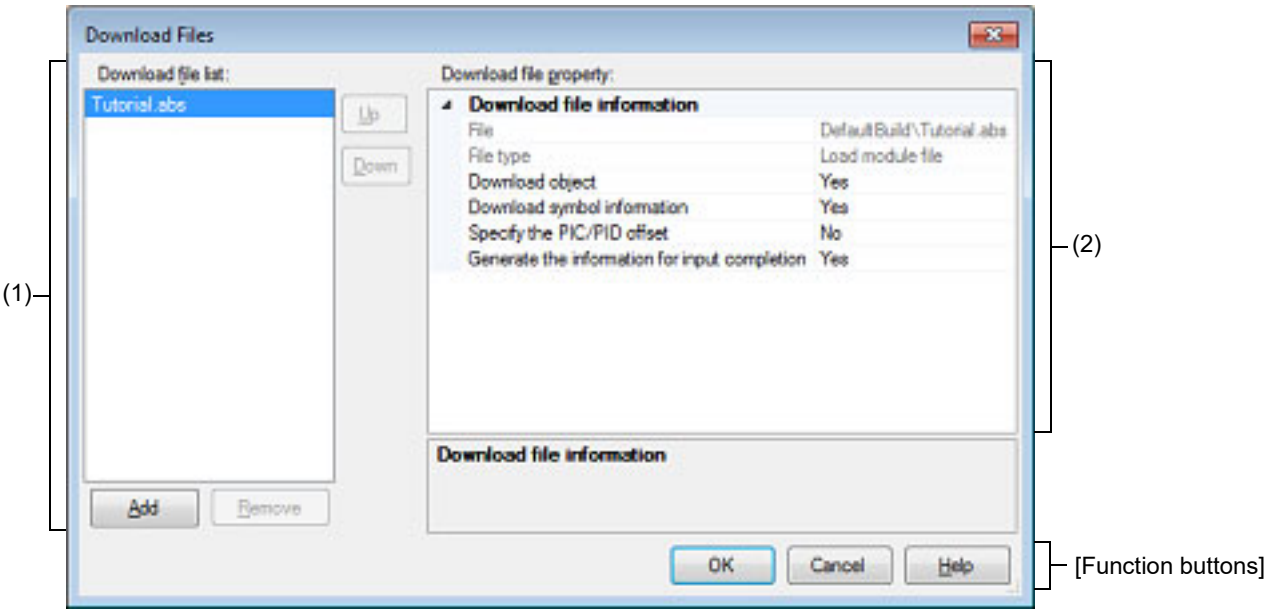
[Function buttons]

Button	Function
OK	Sets a currently set memory mapping in the debug tool and then closes this dialog box.
Cancel	Nullifies changes made to memory mappings and then closes this dialog box.
Help	Displays 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.25 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.

Button	Function
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.

Remark 1. Place the mouse cursor at a file name, and the path information for the subject file is displayed in a popup box.

Remark 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 or S record files. 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	

Generate the information for input completion	This item is displayed only when the files to be downloaded are in the load-module format. Specify whether or not to generate information for use by the auto-complete feature while symbol information is being downloaded from a specified file. If you use the symbol name completion function in "Watch panel", "Memory panel", select is "Yes". Download time is long in order to generate this information during the download.		
	Default	Yes	
	How to change	Select from the drop-down list.	
	Specifiable value	Yes	Generate information for use by the auto-complete feature
		No	Does not generate information for use by the auto-complete feature

Note 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.

Note 2. Unless symbol information is downloaded, source-level debugs cannot be performed.

Note 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]

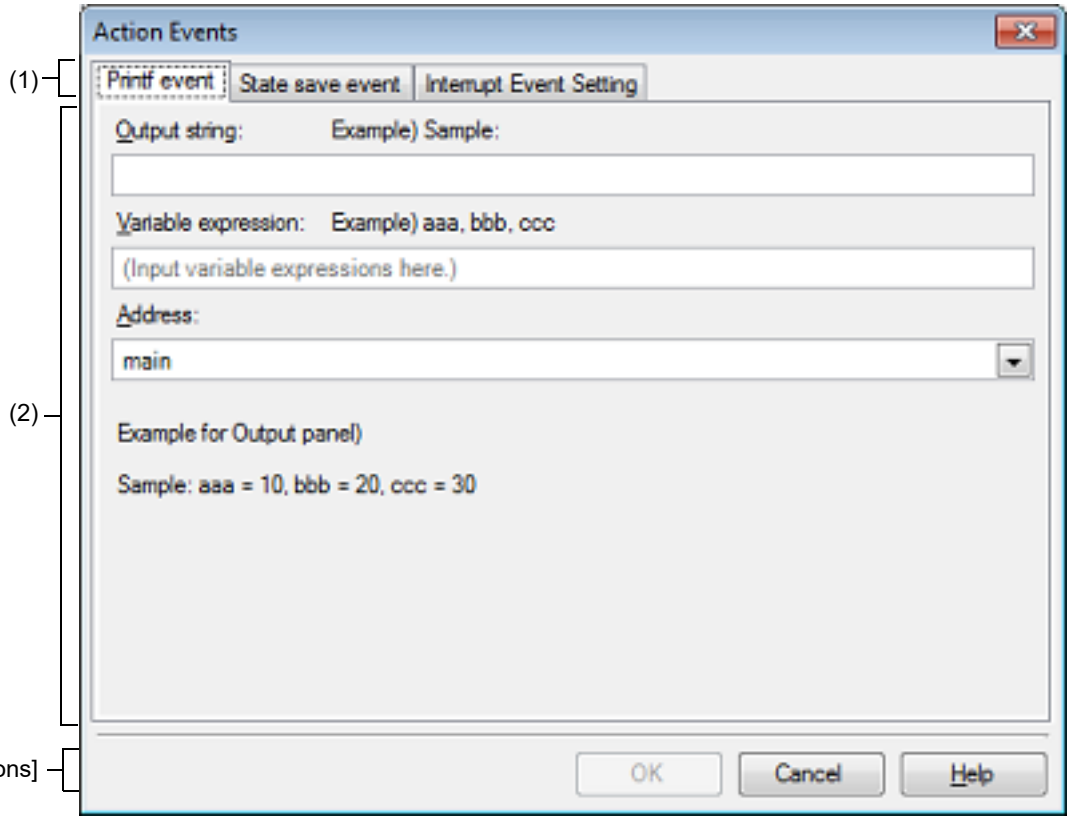
Button	Function
OK	Terminates download file settings and closes this dialog box.
Cancel	Nullifies download file changes made and closes this dialog box.
Help	Displays help for this dialog box.

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, including the allowable number of valid events.

Figure A.26 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 Condition...] 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
- [State save event] tab

- [Interrupt event setting] tab [Simulator]

Caution When you open this dialog box by selecting [Edit Condition...] 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.

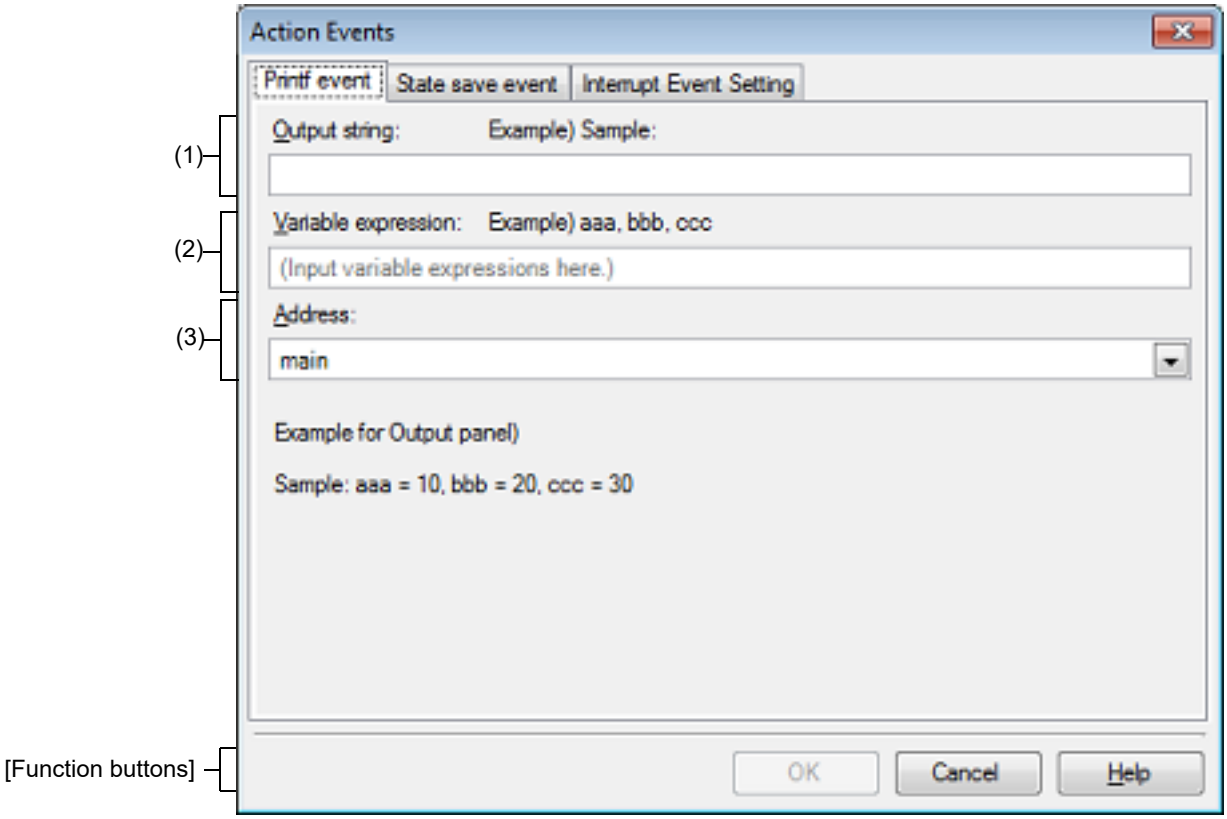
[Function buttons]

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.27 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 Condition...] 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.15 Relationship between Variable Expressions and Output Value (Printf Event)

Variable Expression	Output Value
C/C++ variable name Note 1	Value of a C/C++ variable
Variable expression [*Variable expression*]	Element values of an array
Variable expression.Member name Note 2	Member values of a structure/union/class
Variable expression -> Member name Note 2	Member values of a structure/union/class member pointed to by a pointer
Variable expression.*Cast expression	Value of a pointer to member variable
Variable expression->*Cast expression	Value of a pointer to member variable
**Variable expression*	Value of a pointer variable
&*Variable expression*	Location address
(Type name) *Variable expression*	Value cast to a specified type
CPU register name	Value of a CPU register
I/O register name	Value of an I/O register
Label name Note 3, EQU symbol name Note 3, [immediate value]	Value of a label, value of an EQU symbol, a value of an immediate address

Note 1. C89, C99, or C++ language variable

Note 2. When using a member variable of a base class, specify the scope before the member name (e.g. variable.BaseClass::member).

Note 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 "2.21.2 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 "2.21.2 Symbol name completion function").

Note that the output result format by the Printf event in the Output panel are as follows:

Figure A.28 Output Result Format of Printf Event

```
Specified characters  Variable expression 1 = Value 1, Variable expression 2 = Value 2,
Variable expression 3 = Value 3, ...
```

Specified characters	Characters specified with [Output string]
Variable expression 1 - 10	Characters specified with [Variable expression]
Value 1 - 10	Value of variable corresponds to *" Variable expression 1 - 10 "* The value is displayed in a format that matches the variable type (see Table A.11 Display Form 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.)

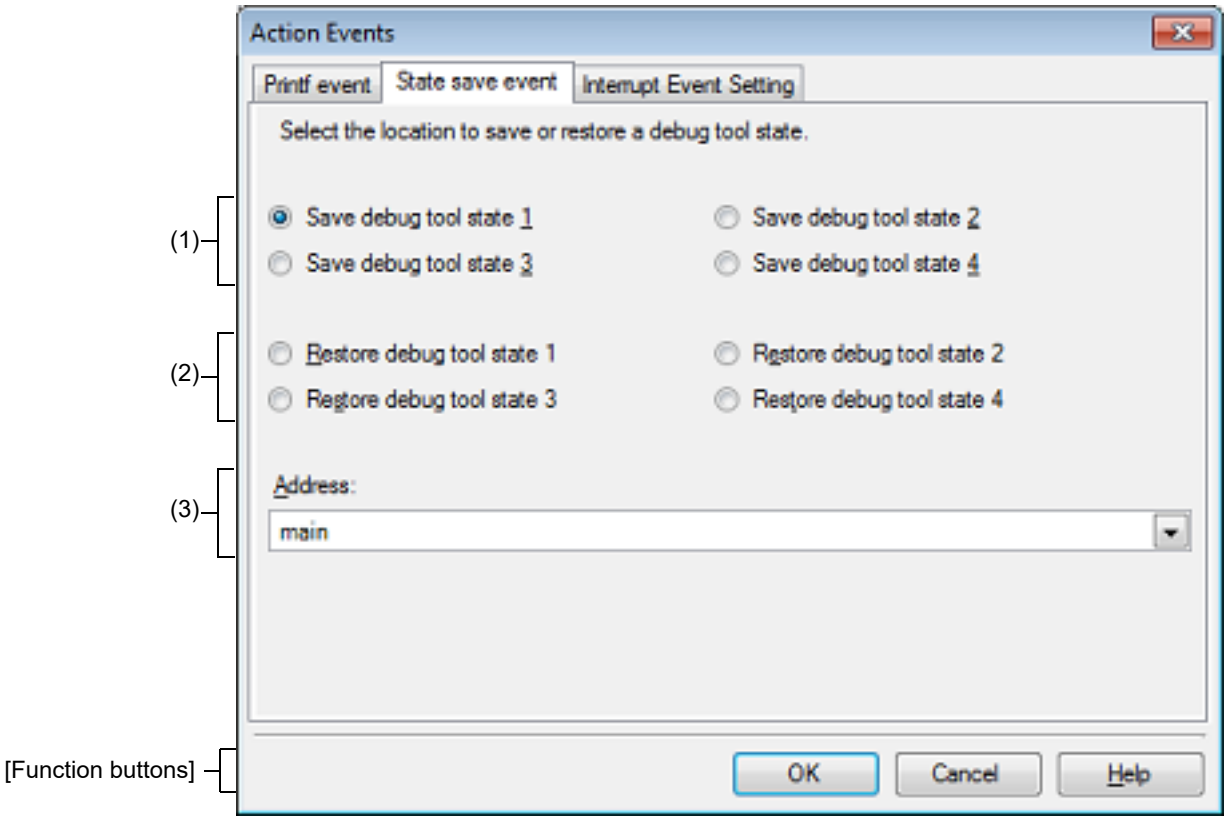
[Function buttons]

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.

[State save event] tab

This tab is used to configure how to save and restore the state of the debug tool upon occurrence of an action event. Note that the data to be restored is limited to memory and register values that can be read or written.

Figure A.29 Action Events Dialog Box: [State save 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 state save 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 state save event, then select [Register Action Event...] from the context menu.
- On the Events panel, select [Edit Condition...] from the context menu after selecting the state save event.

[Description of each area]

(1) [Save debug tool state *n*] area
When an action event occurs, the state of the debug tool is saved in a file as the *n*-th data.

(2) [Restore debug tool state *n*] area
When an action event occurs, the state of the debug tool is restored from the *n*-th data file.

(3) [Address] area
Specify the address at which to set a state save event.

You can either type an address expression directly into the text box (up to 1024 characters), or select them from the input history via the drop-down list (up to 10 items). The address of the location currently being specified is displayed by default.

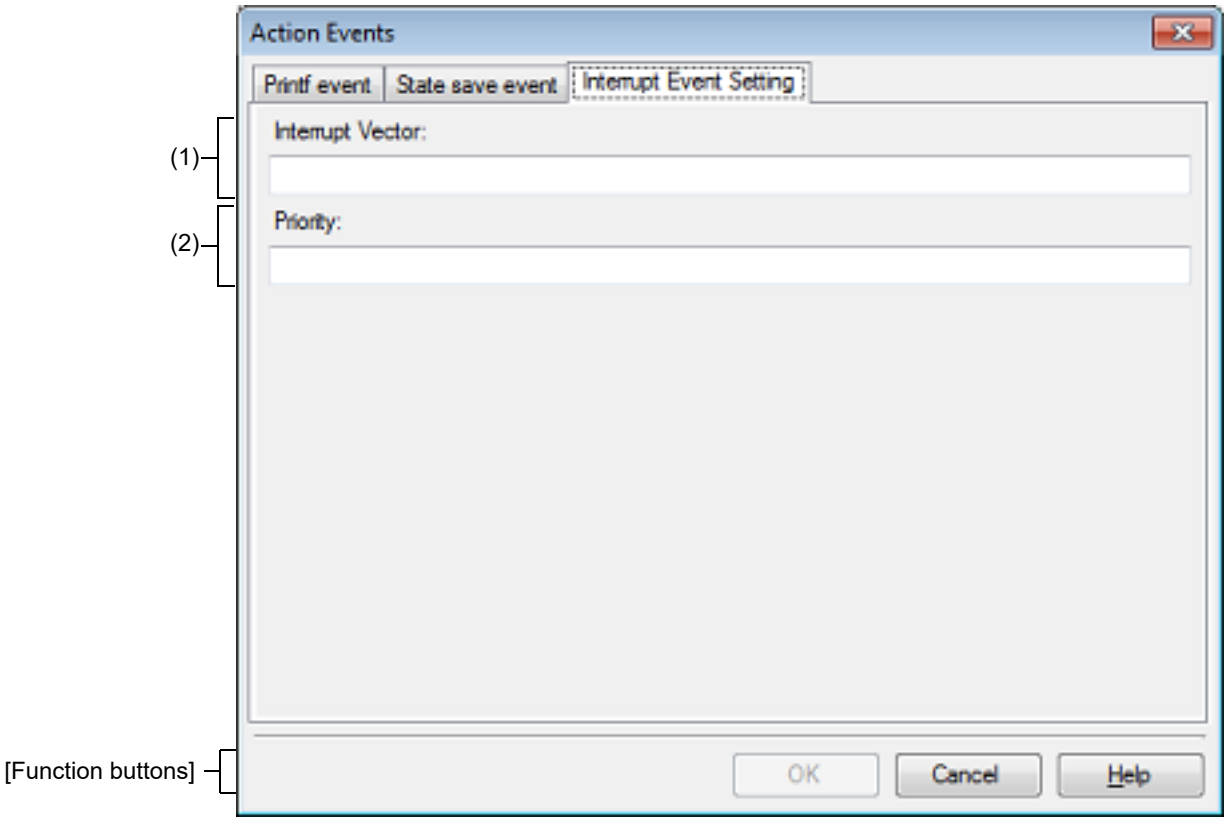
[Function buttons]

Button	Function
OK	Finishes configuring the state save event, and sets it to the line/address at the caret position in the Editor panel/Disassemble panel.
Cancel	Cancels the state save 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.30 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 Condition...] 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.

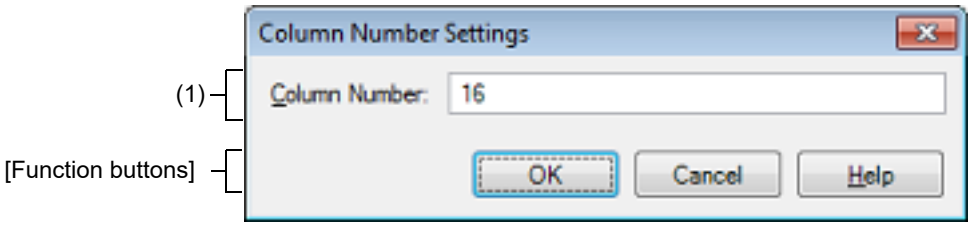
[Function buttons]

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.

Column Number Settings dialog box

This dialog box is used to set the number of view columns of memory values on the Memory panel.

Figure A.31 Column Number Settings Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Memory panel, select [View] >> [Column Number Settings…] from the context menu.

[Description of each area]

(1) [Column Number] area
Directly enter a decimal value as the number of columns you want to display.
The settable range depends on [Size Notation] currently being set on the Memory panel, as follows:

Size Notation	Settable Range
4 Bits	2 - 512[Note]
1 Byte	1 - 256
2 Bytes	1 - 128
4 Bytes	1 - 64
8 Bytes	1 - 32

Note Only an even number is specifiable (if an odd number is specified, then it will be changed to a value one greater than such odd number).

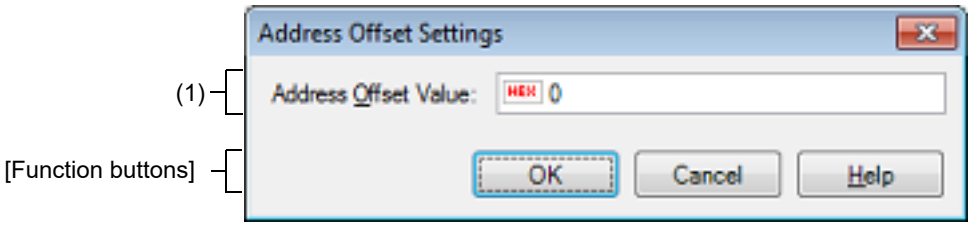
[Function buttons]

Button	Function
OK	Displays memory values in the specified number of columns.
Cancel	Nullifies settings and then closes this dialog box.
Help	Displays the help for this dialog box.

Address Offset Settings dialog box

This dialog box is used to set an offset value of the start address in the address area on the Memory panel.

Figure A.32 Address Offset Settings Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Memory panel, select [View] >> [Address Offset Value Settings...] from the context menu.

[Description of each area]

(1) [Address Offset Value] area
Directly enter a hexadecimal value as an offset value for the address display.
The settable range depends on the number of bytes of the memory currently being displayed in a line on the Memory panel, as follows:

- Settable range: 0x0 - ("*Set value of [Size Notation]*" x "*The number of view columns*") -1

Example When "*Set value of [Size Notation]*" is 1 byte and "*The number of view columns*" is 16 columns:

Offset Value	Displayed Content of Address Area
0x0 (default)	0000 0010 0020
0x1	0001 0011 0021
0x2	0002 0012 0022

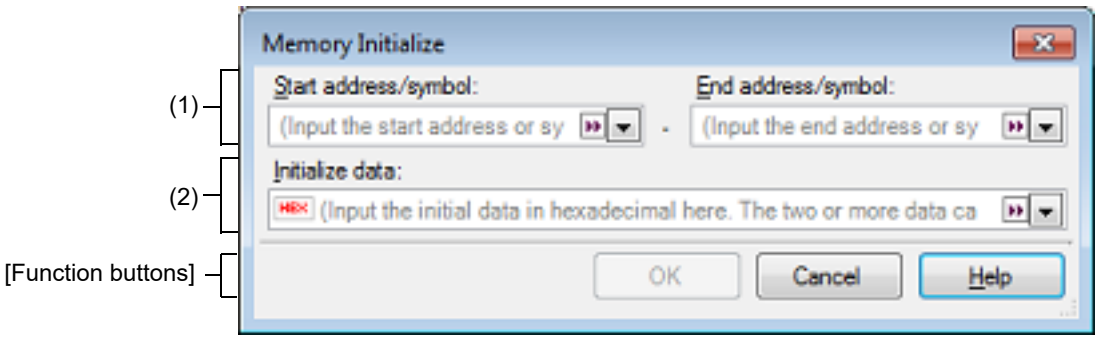
[Function buttons]

Button	Function
OK	Displays memory addresses with the specified offset value.
Cancel	Nullifies settings and then closes this dialog box.
Help	Displays the help for this dialog box.

Memory Initialize dialog box

This dialog box initializes memory values. (See "2.11.1.6 Collectively changing (initializing) memory contents".)
A pattern of specified initialization data is written into a specified address range of memory repeatedly.

Figure A.33 Memory Initialize Dialog Box



Here, the following items are explained.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Choose [Fill...] 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 "2.21.2 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

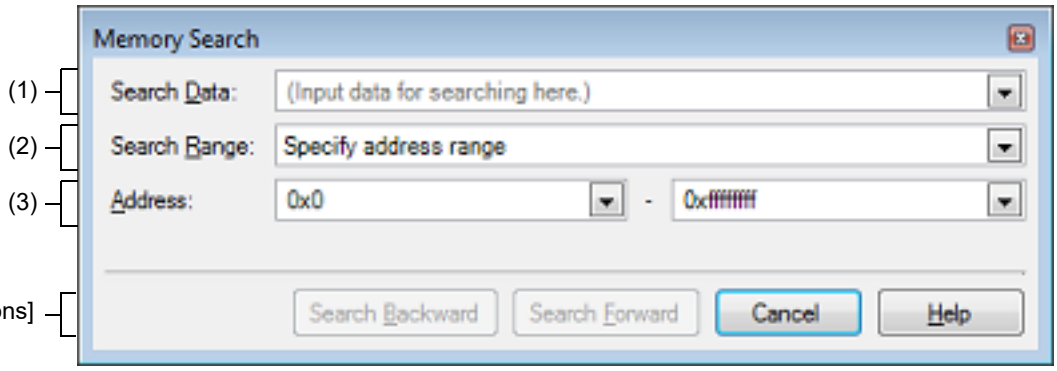
[Function buttons]

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 "2.11.1.5 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.34 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 encoding 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.
Memory mapping	A search is conducted within the selected range of memory mapping. This list item displays memory mappings individually (except non-mapped areas) that are displayed in the Memory Mapping dialog box. Display form: *<Memory type> <Address range> <Size>*

(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.

Remark 1. By holding down [Ctrl]+[Space] keys in this text box, you can complete the symbol name at the present caret position (see "2.21.2 Symbol name completion function").

Remark 2. If the "start address" text box is blank, address "0x0" is assumed.

Remark 3. If the "end address" text box is blank, the upper-limit address of the microcontroller's address space is assumed.

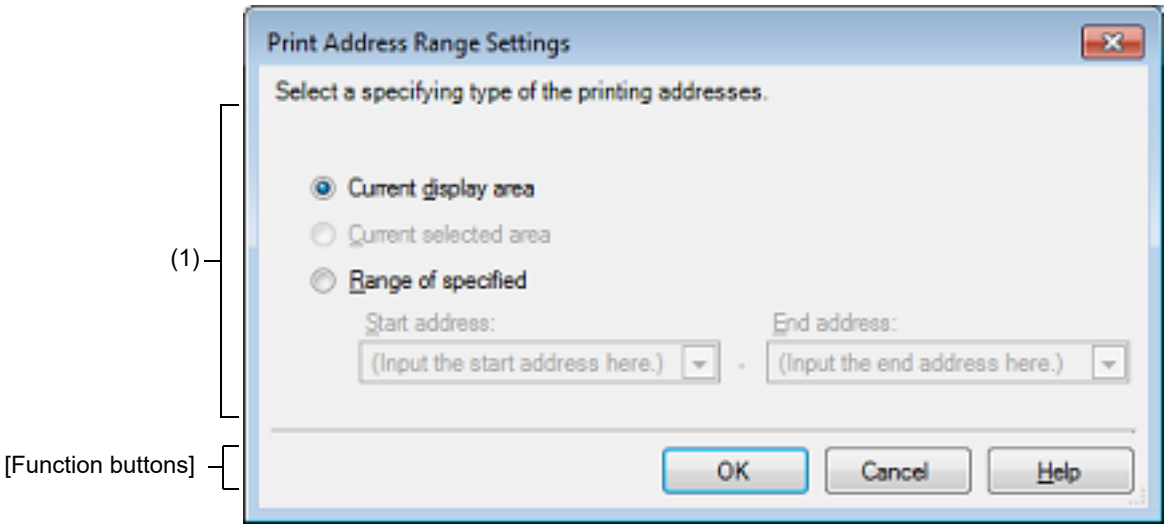
[Function buttons]

Button	Function
Search Backward	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. 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. 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.
Search Forward	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. 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. 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.
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

When printing the content of the Disassemble panel, specify an address range of the subject to be printed.

Figure A.35 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, choose [Print...] from the [File] menu.

[Description of each area]

(1) Range specification area
To specify a range to be printed, select one of the option buttons given below.

(a) [Current display area] (default)
Prints only a currently displayed range on the Disassemble panel.

(b) [Current selected area]
Prints only a currently selected range on the Disassemble panel.
However, this is disabled if nothing is selected on the Disassemble panel.

(c) [Range of specified]
Specify an address range of the subject to be printed by [Start address] and [End address].
Enter an address expression 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).

Remark By pressing the [Ctrl] + [Space] keys in this text box, it is possible to complement a symbol name at the current caret position (see "2.21.2 Symbol name completion function").

[Function buttons]

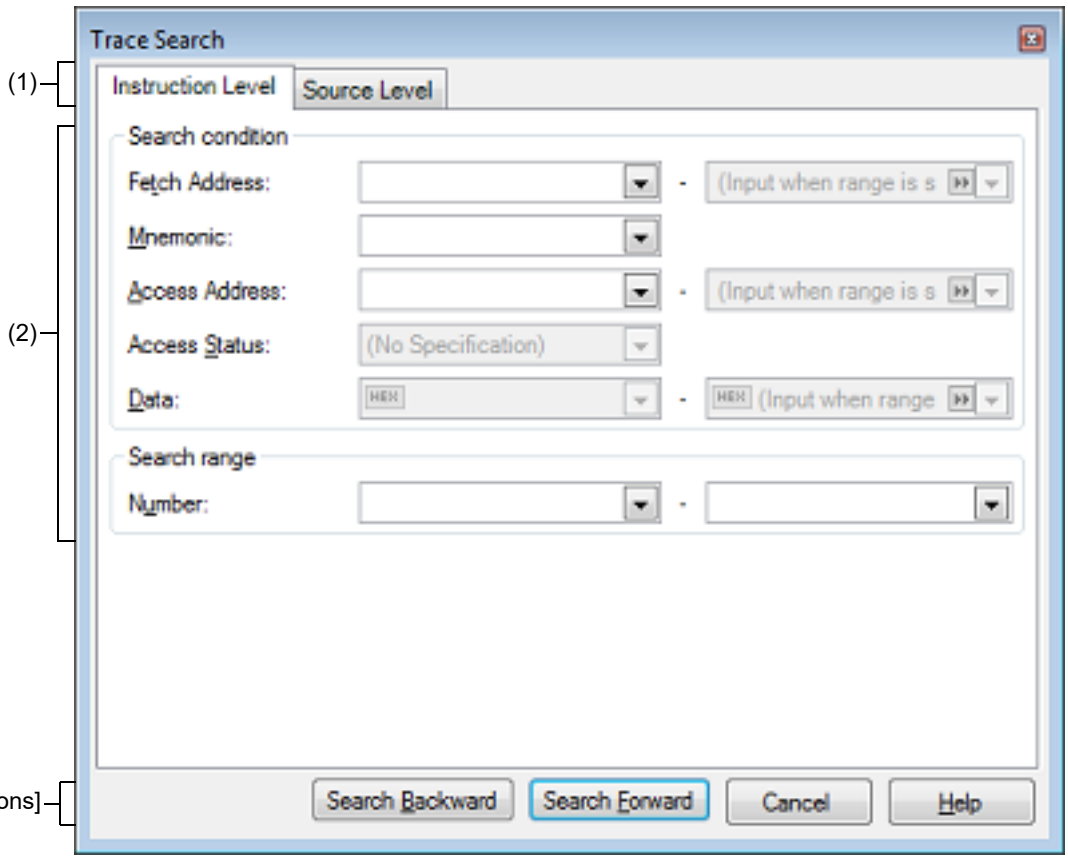
Button	Function
OK	Closes this dialog box and opens a Windows printing dialog box to print a specified range of contents of the Disassemble panel.

Button	Function
Cancel	Closes this dialog box ignoring range selection settings.
Help	Displays help for this dialog box.

Trace Search dialog box

This dialog box searches for trace data (see Section "2.13.8 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.36 Trace Search Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Click 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]

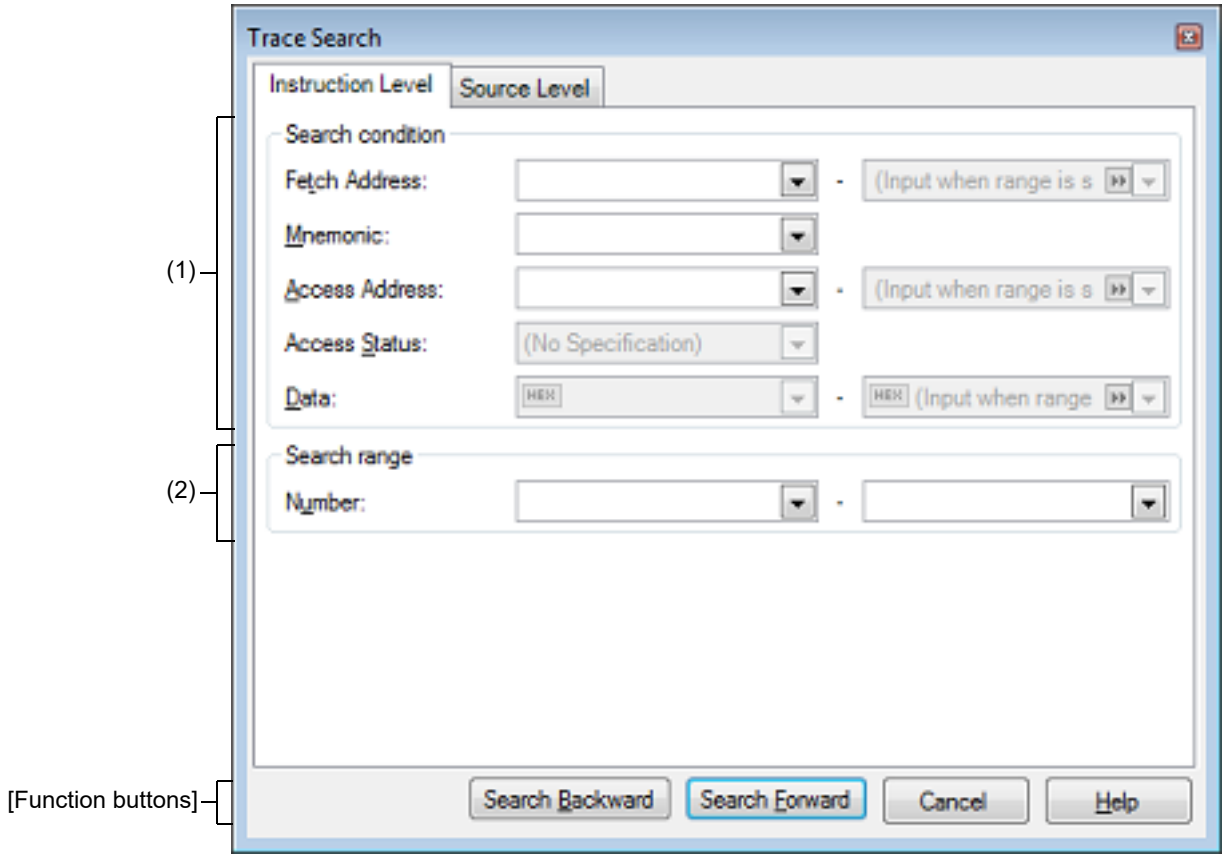
Button	Function
Search Backward	Performs a search in the direction toward smaller addresses within a specified range. The searched spot is put in selected state on the Trace panel. However, if an invalid value is specified or when the program is under execution, a message is displayed and a search of 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 or when the program is under execution, 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.

[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.37 Trace Search Dialog Box: [Instruction Level] Tab



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Click 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
Vector Read **[Simulator]**
DMA

Caution [E1] [E20] [EZ Emulator]
The types of access for which data can be collected on the Trace panel are Read and Write.
Therefore, do not select Read/Write, Vector Read or DMA from the drop-down list.
[Simulator]
The types of access for which data can be collected on the Trace panel are only Read, Write, and Vector Read. Therefore, do not select Read/Write or DMA from the drop-down list.

(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]

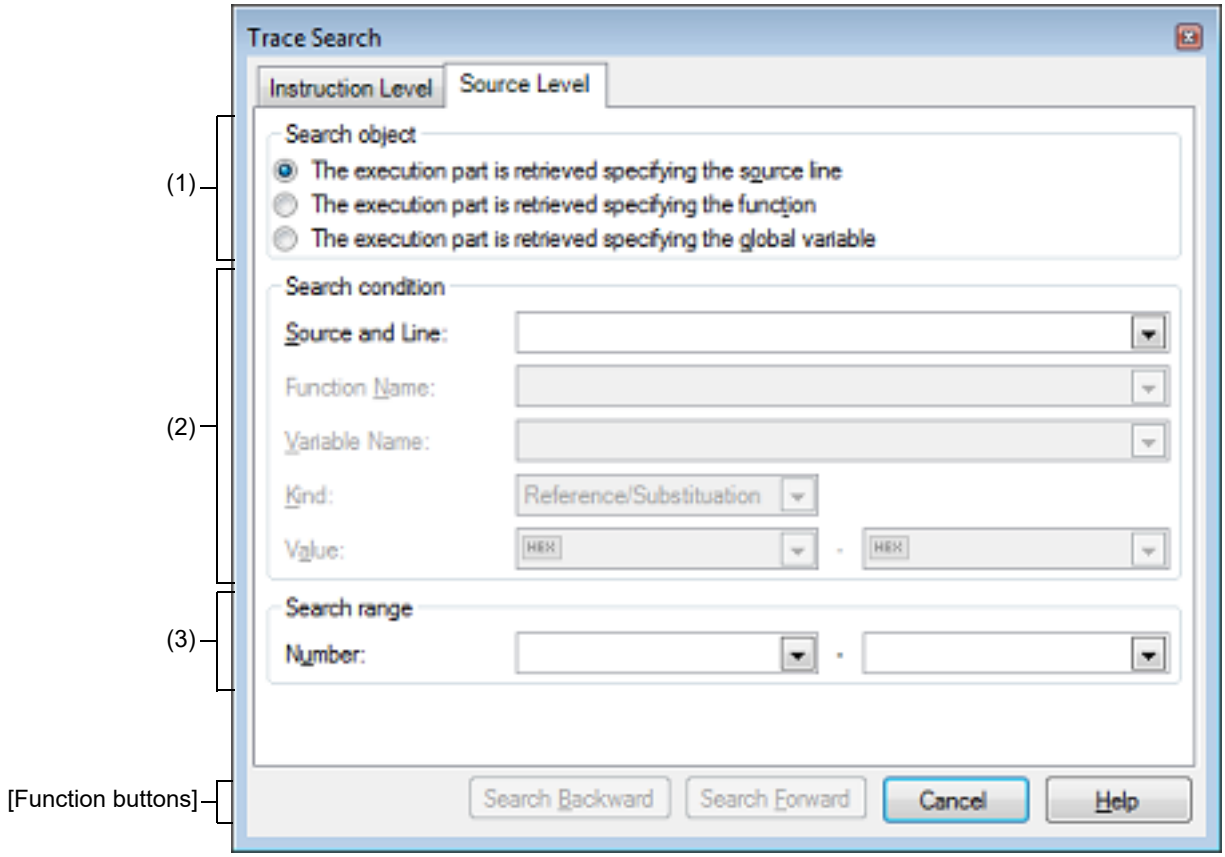
Button	Function
Search Backward	Performs a search in the direction toward smaller addresses within a specified range. The searched spot is put in selected state on the Trace panel. However, if an invalid value is specified, a message is displayed and a search of trace data is not performed. Also, if 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.38 Trace Search Dialog Box: [Source Level] Tab



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Click 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.

Example 1. main.c#40

Example 2. main.c

Example 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.

Button	Function
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] [EZ Emulator]

This dialog box is used to change the detailed information on combination condition for the combination break or for the 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.39 Combination Condition Dialog Box (Combination Break)

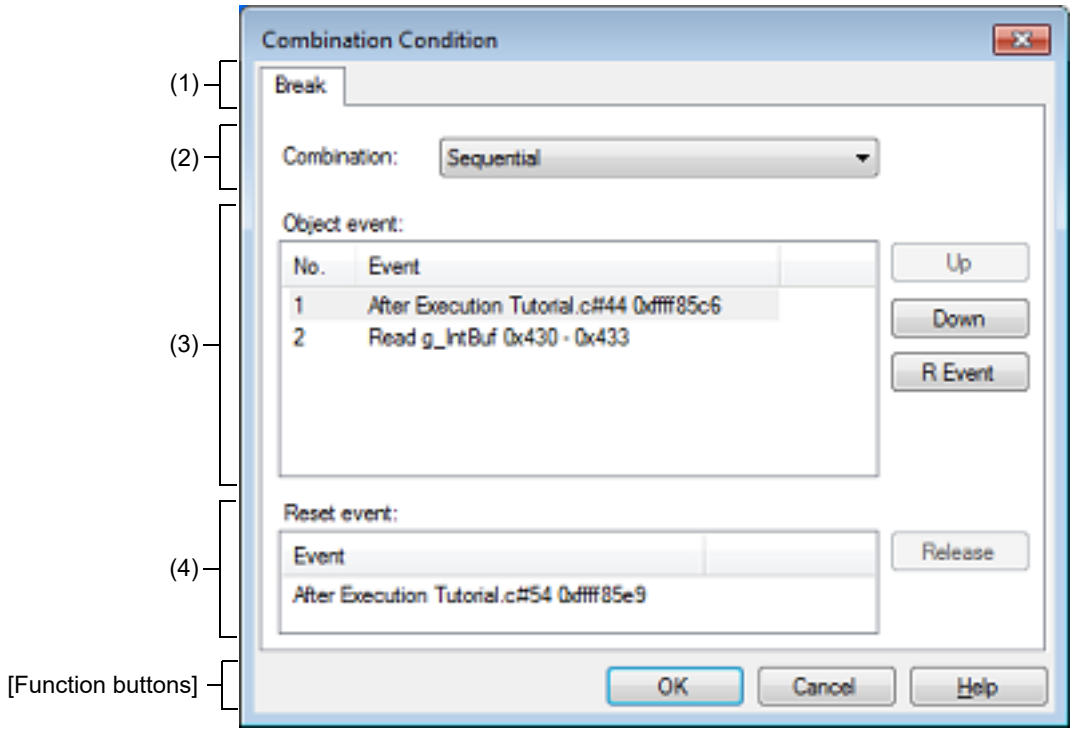
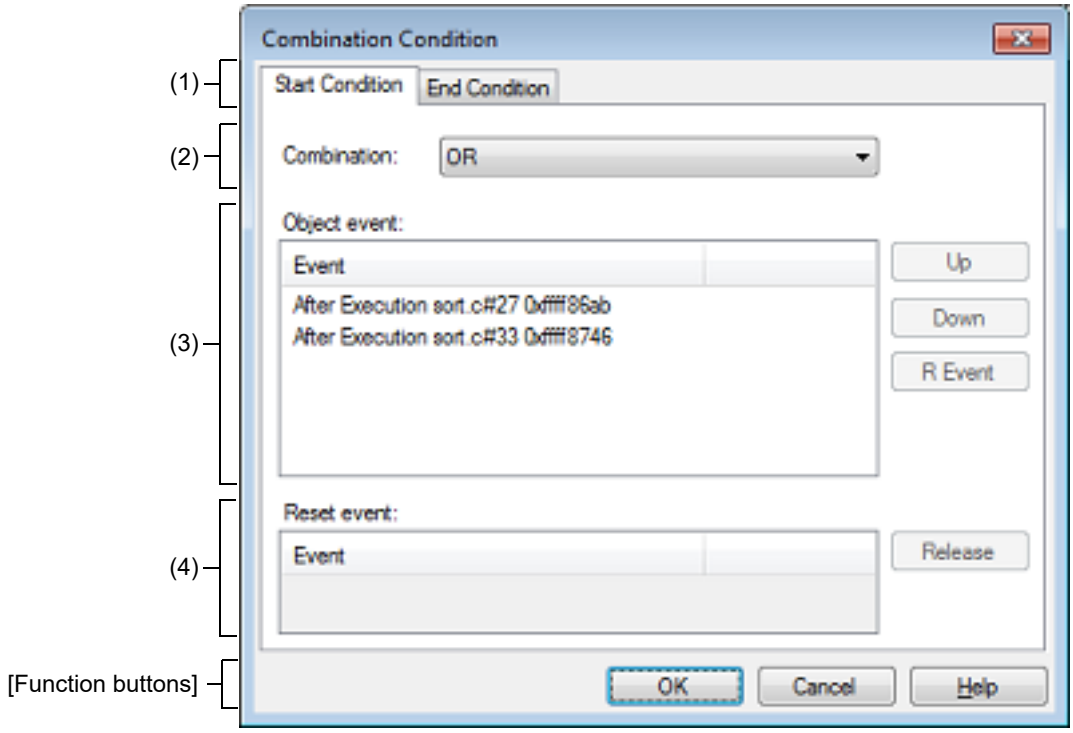


Figure A.40 Combination Condition Dialog Box (Trace Event)



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 of which you wish to change the detailed information, then select [Edit Condition...] from the context menu.
- On the Events panel, move the caret to the trace of which you wish to change the detailed information, then select [Edit Condition...] from the context menu.

[Description of each area]

(1) Tab selection area
You can display the detailed information on 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 [End 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.

Caution Also see Cautions of "(1) Editing the combination condition" for details on combination conditions.

(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 Object event list. This button is enabled only when you have specified [Sequential] for the combination conditions.

Buttons	Functions
Down	Moves the event serial number downward in the Object event list. This button is enabled only when you have specified [Sequential] for the combination conditions.
R Event	Moves the event selected in the Object 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.

Caution Also see Cautions of "(3) Editing the reset event" for details on Editing the reset event.

(b) Button

Button	Function
Cancel	Moves the event inside the Reset event list to the Object event list. This button is enabled only when you have specified [Sequential] for the combination conditions.

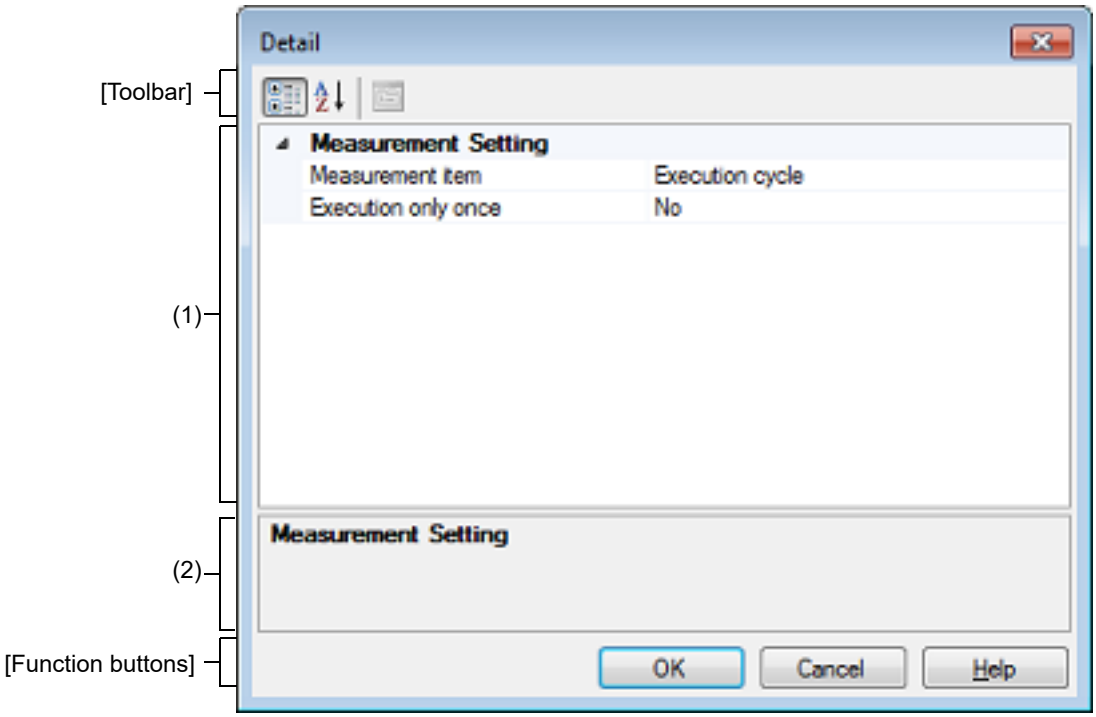
[Function buttons]

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] [EZ Emulator]

This dialog box is used to display and change 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.41 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 of which you wish to change the detailed information, 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 Note	- RX600, RX700 Series Specify the measurement item. - RX200 Series, RX140Group Displays "Execution cycle".		
	Default	*Execution cycle*	
	Modifying	Select from the drop-down list	
	Available values	One of the following as selected from the drop-down list. [RX600, RX700 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, RX140Group] *Execution cycle*	
Execution only once	Specifies whether to end the measurement after measuring the specified segment only once.		
	Default	*No*	
	Modifying	One of the following as selected 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.

Note [RX100 Series other than RX140 Group]
These microcontrollers does not support timers.

[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

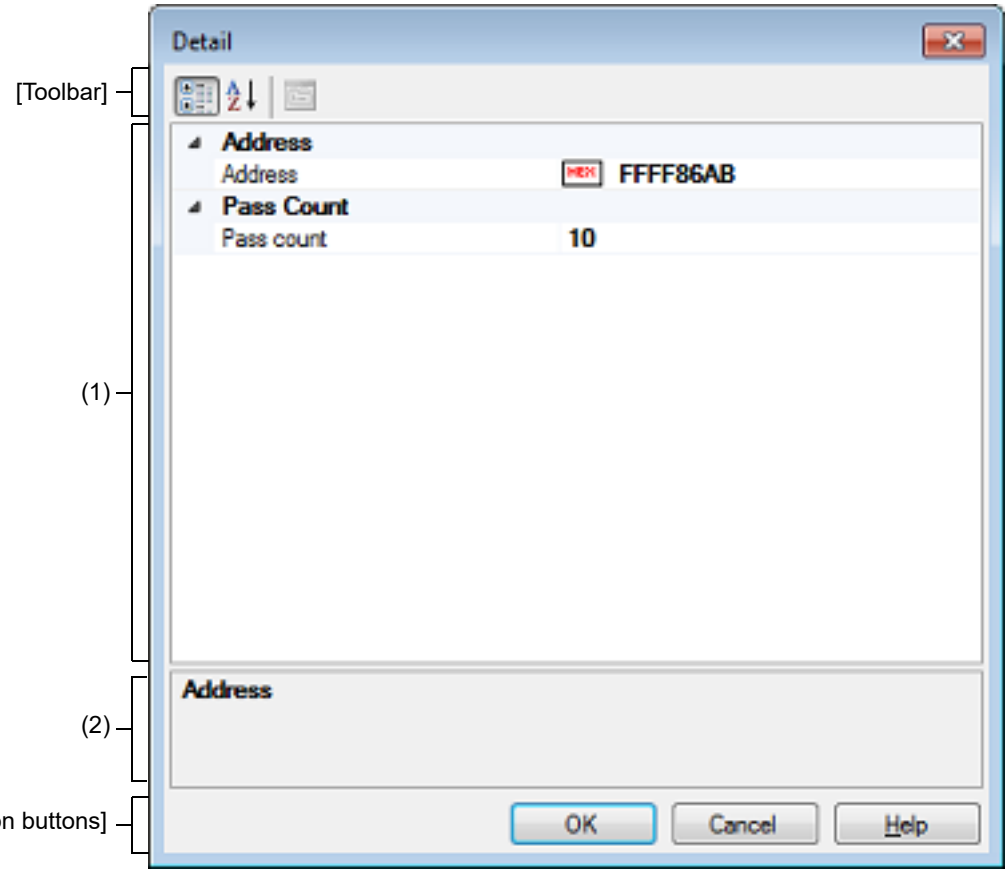
This dialog box is used to display and change detailed information on execution-related events selected on the Events panel.

Note that the execution-related events refer to the following events on the Events panel.

- Hardware break
- After execution (or post-execution) break in the detailed information on combination break **[E1] [E20] [EZ Emulator]**
- After execution (or post-execution) events[Note] registered as trace start and trace end events in the detailed information on trace
- After execution (or post-execution) events[Note] registered as timer start and timer end events in the detailed information on timer result (or timer measurement)

Note **[Simulator]**
Before execution (or pre-execution) events apply instead of after execution events.

Figure A.42 Detailed Settings of Execution Events 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 hardware break of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.
- On the Events panel, move the caret to the execution-related event in trace of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.
- On the Events panel, move the caret to the execution-related event in timer result of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.

[E1] [E20] [EZ Emulator]

- On the Events panel, move the caret to the execution-related event in combination break of which you wish to change the detailed information, 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	Display and specify the address at which an execution-related event is set.	
	Default	*Depends on selected event*
	modifying	By entering directly from the keyboard
	Available values	0x0 to 0xFFFFFFFF

(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] [EZ Emulator] 1 to 256 - [Simulator] 1 to 16,383

Caution 1. [E1] [E20] [EZ Emulator]
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.

Caution 2. [E1(Serial)/E20(Serial)/EZ Emulator [RX100, 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 is used to display or change detailed information on access-related events 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 the detailed information on combination break **[E1] [E20] [EZ Emulator]**
- Read, write, and read/write in the detailed information on hardware break **[Simulator]**
- Read, write, and read/write in the detailed information on trace
- Read, write, and read/write in the detailed information on point trace
- Read, write, and read/write in the detailed information on timer result

Figure A.43 Detailed Settings of Access Events Dialog Box [E1] [E20] [EZ Emulator]

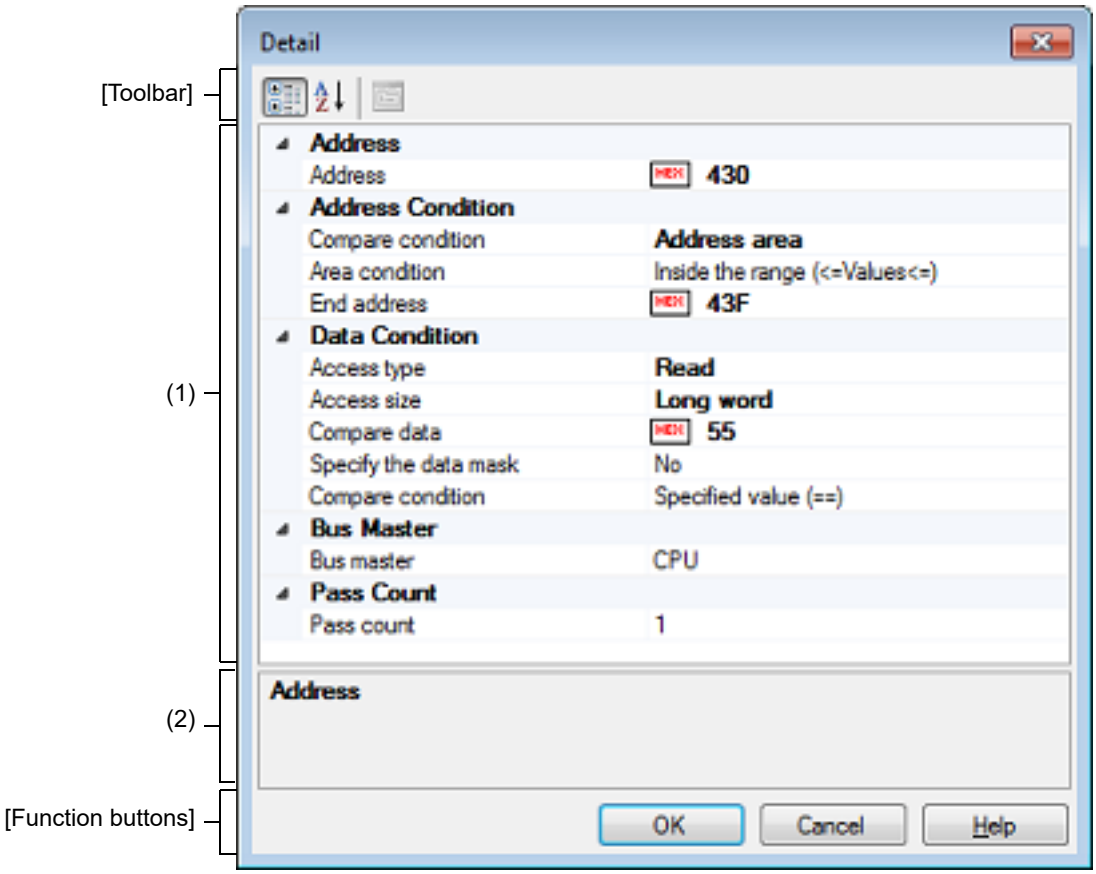
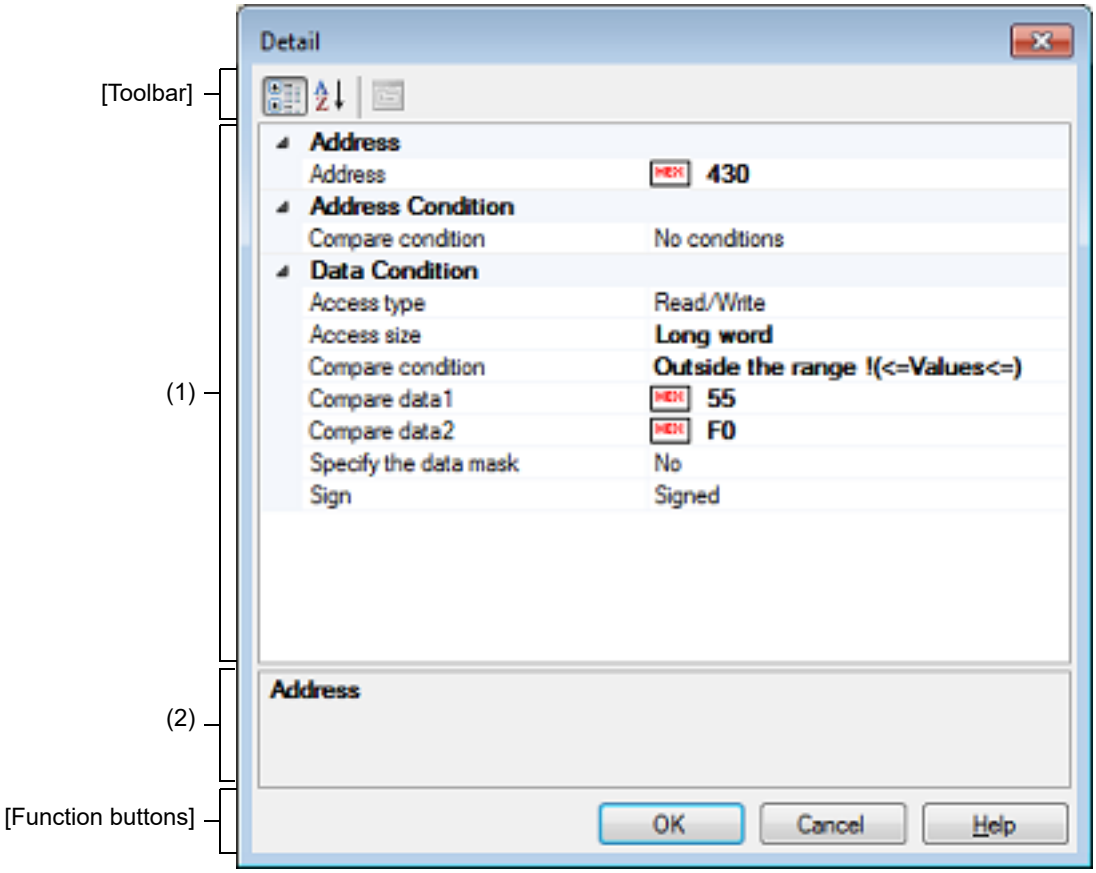


Figure A.44 Detailed Settings of Access Events Dialog Box [Simulator]



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 the access-related event in trace of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.
- On the Events panel, move the caret to the access-related event in point trace of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.
- On the Events panel, move the caret to the access-related event in timer result of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.

[E1] [E20] [EZ Emulator]

- On the Events panel, move the caret to the access-related event in combination break of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.

[Simulator]

- On the Events panel, move the caret to the access-related hardware break of which you wish to change the detailed information, 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] [EZ Emulator]
(2) For [Simulator]

(1) For [E1] [E20] [EZ Emulator]

(a) [Address]

Address	Display and specify the address at which an access-related event is set.	
	Default	*Depends on selected event*
	modifying	By entering directly from the keyboard
	Available values	0x0 to 0xFFFFFFFF

(b) [Address Condition]

Compare condition	Specify address comparison condition.		
	Default	*No specification*	
	Modifying	By selecting from the drop-down list	
	Available values	No conditions	No address comparison condition is specified
		Address area[Note1 Note2]	Specify comparison condition based on an address range.
		Compare with address mask	Specify masked comparison condition.
Area condition	Specify address comparison range condition. This property is displayed only when [Address area] 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 comparison 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 (==)	Condition holds true when the address range that matches the result of the mask is accessed.
		Any other value (!=)	Condition holds true when any address that does not match the result of the mask is accessed.

Note 1. [RX100, RX200 Series]
Comparison conditions based on an address area are not supported.

Note 2. [RX600, RX700 Series]
You can specify "Address area" as the comparison condition only to one event.

Note 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	Specify mask data in hexadecimal.[Note] 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 comparison 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 (==)	Condition holds true with the access of the data value that matches the result of the mask.
		Any other value (!=)	Condition holds true with any access of the data value that does not match the result of the mask.

Note The data value to be used as the condition is masked bit-wise with a mask value for which "0" is specified.

(d) [Bus master] [RX71M, RX66x, RX65x, RX64M, RX66T, RX671, RX72T, RX72M, and RX72N Groups]

Bus master	Specify the bus master.		
	Default	*CPU*	
	How to change	By selecting from the drop-down list	
	Specifiable value	CPU	When the specified data access from the CPU occurs, the condition holds true.
		DMAC/ DTC	When the specified data access from the DMAC/DTC occurs, the condition holds true.

Caution For an microcontroller that does not have the function for selecting the Bus master of data access, [Bus master] property is not displayed.

(e) [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

Caution 1. [E1] [E20] [EZ Emulator]
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.

Caution 2. [E1(Serial)/E20(Serial)/EZ Emulator [RX100, 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	Display and specify the address at which an access-related event is set.	
	Default	*Depends on selected event*
	modifying	By entering directly from the keyboard
	Available values	0x0 to 0xFFFFFFFF

(b) [Address Condition]

Compare condition	Displays address comparison condition.	
	Default	*No conditions*
	Modifying	Not changeable

(c) [Data Condition]

Access type	Specify the type of access.		
	Default	*Depends on selected event*	
	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	*Depends on selected event*	
	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 comparison condition. This property is not displayed when [No conditions] is specified in the [Access size] property.		
	Default	*Equal (==)*	
	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. - 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. - 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. - 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. - 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.

Note 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.

Note 2. The address value to be used as the condition is masked bit-wise with a mask value for which "0" is specified.

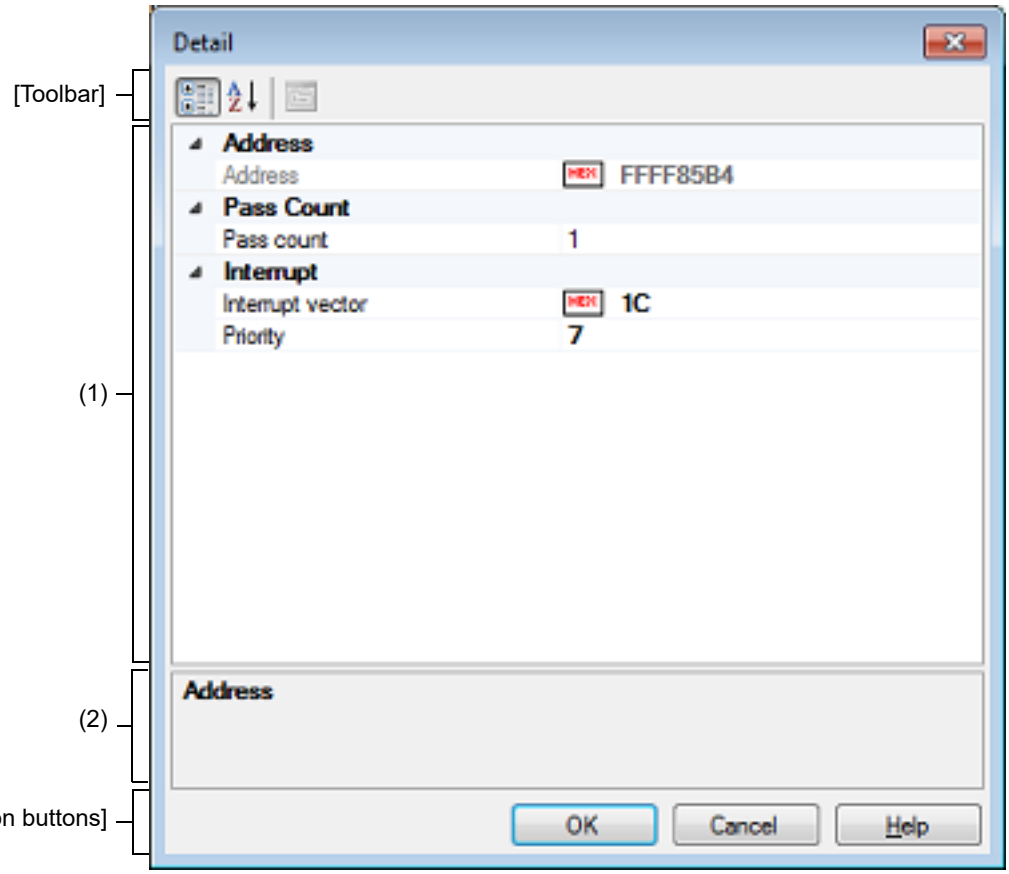
[Function buttons]

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 is used to display and change detailed information on an interrupt event selected on the Events panel.

Figure A.45 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 the interrupt event of which you wish to change the detailed information, 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	*Depends on the selected event*
	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	*Depends on the selected event*
	Modifying	Direct entry from the keyboard
	Specifiable value	0x00 to 0xFF
Priority	Specifies an interrupt's priority level.	
	Default	*Depends on the selected event*
	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.

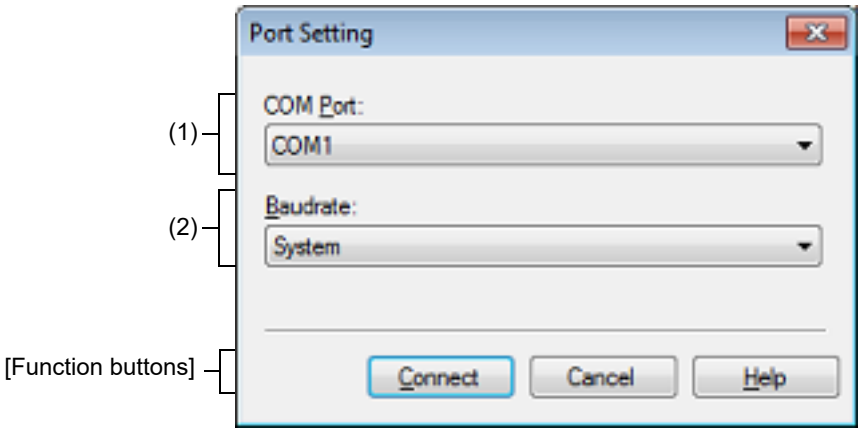
[Function buttons]

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.46 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.

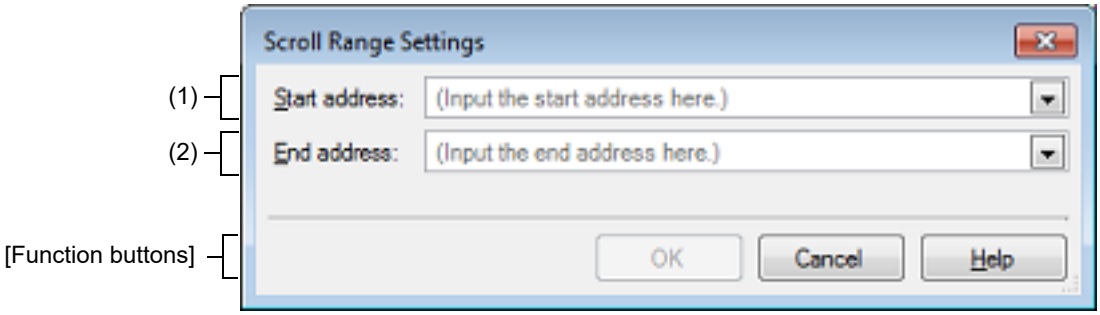
Scroll Range Settings dialog box

Set a range in which the vertical scroll bar of the Memory panel or Disassemble panel is scrolled.
When an appropriate range is set, the slider on the vertical scroll bar of the panel changes in size, so that dragging or the like operation with the mouse will be improved.

Caution Even when, after a scroll range is set in this dialog box, the addresses represented by specified address expressions are altered by execution of a line assemble, etc., the scroll range is not corrected.

Remark Movement in a panel window by the [Page Up], [Page Down], [Up] or [Down] key or a button at the end of the scroll bar, or by selection of a jump-related menu is possible, even outside the scroll range.

Figure A.47 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, select [View] and then click the button in the toolbar.
- On the Memory panel, select [View] and then [Settings Scroll Range...] on the context menu.
- On the Disassemble panel, select [View] and then click the button in the toolbar.
- On the Disassemble panel, select [View] and then [Settings Scroll Range...] on the context menu.

[Description of each area]

(1) [Start address] area
Specify the start address of a range to be scrolled.
Enter an address expression directly in the text box (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries).
Note that if "All" in the drop-down list is selected, no scroll ranges are set (no limits to scroll range).

Remark By pressing the [Ctrl] + [Space] keys in this text box, it is possible to complement a symbol name at the current caret position (see "2.21.2 Symbol name completion function").

(2) [End address] area
Specify the end address of a range to be scrolled.
Enter an address expression directly in the text box (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries).
However, this area disabled if "All" is specified in [Start address]].

Remark By pressing the [Ctrl] + [Space] keys in this text box, it is possible to complement a symbol name at the current caret position (see "2.21.2 Symbol name completion function").

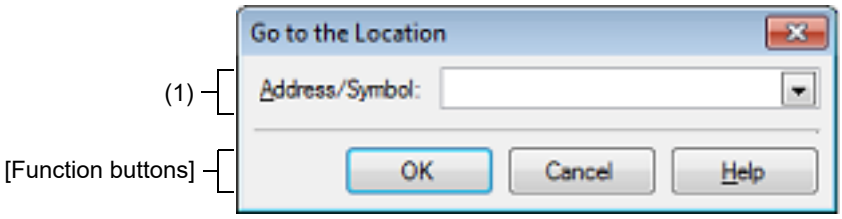
[Function buttons]

Button	Function
OK	Sets a specified scroll range in the panel concerned with it and moves the caret to the subject panel, with the start address appearing at the beginning of display.
Cancel	Nullifies settings and then closes this dialog box.
Help	Displays help for this dialog box.

Go to the Location dialog box

Move the caret to a specified position.

Figure A.48 Go to the Location Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Choose [Go To…] from the [Edit] menu.
- On the Disassemble panel, select [Go To…] on the context menu.
- On the IOR panel, select [Go To…] on the context menu.

[Description of each area]

(1) [Address/Symbol] or [IOR] area
Specify a location to which you want the caret to be moved.
Enter directly in the text box (specifiable in up to 1,024 characters) or select an input history item from the drop-down list (up to 10 history entries).
The content of specification differs with the panel concerned, as shown below.

Subject panel	Content of specification
Disassemble panel	Address expression
IOR panel	I/O register name

Remark If this dialog box is opened from the Disassemble panel by pressing the [Ctrl] + [Space] keys in the text box here, it is possible to complement a symbol name at the current caret position (see "2.21.2 Symbol name completion function").

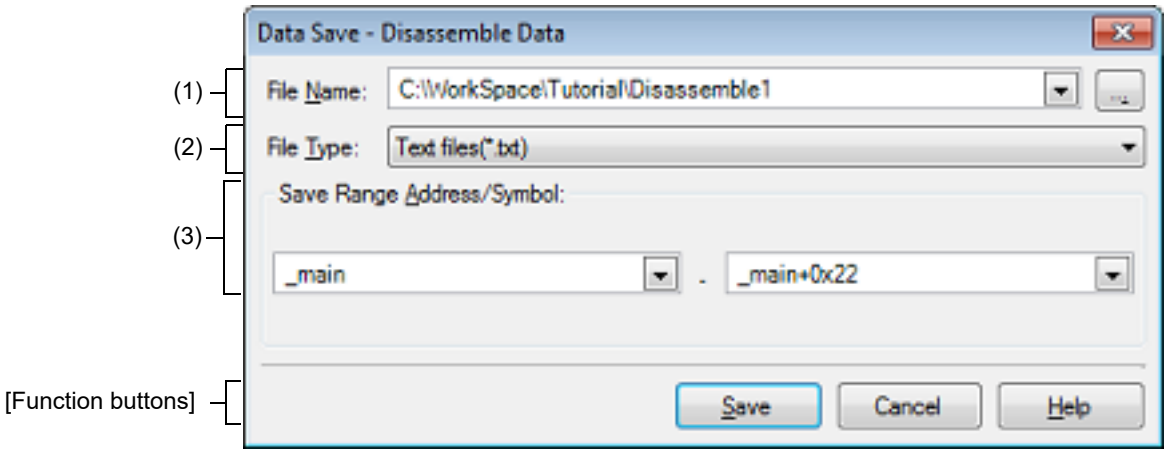
[Function buttons]

Button	Function
OK	Moves the caret to the subject panel, with a specified position appearing at the beginning of display.
Cancel	Nullifies settings and then closes this dialog box.
Help	Displays help for this dialog box.

Data Save dialog box

This dialog box 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 box can only be opened when CS+ is connected with the debug tool.

Figure A.49 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	- 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 "2.21.2 Symbol name completion function").

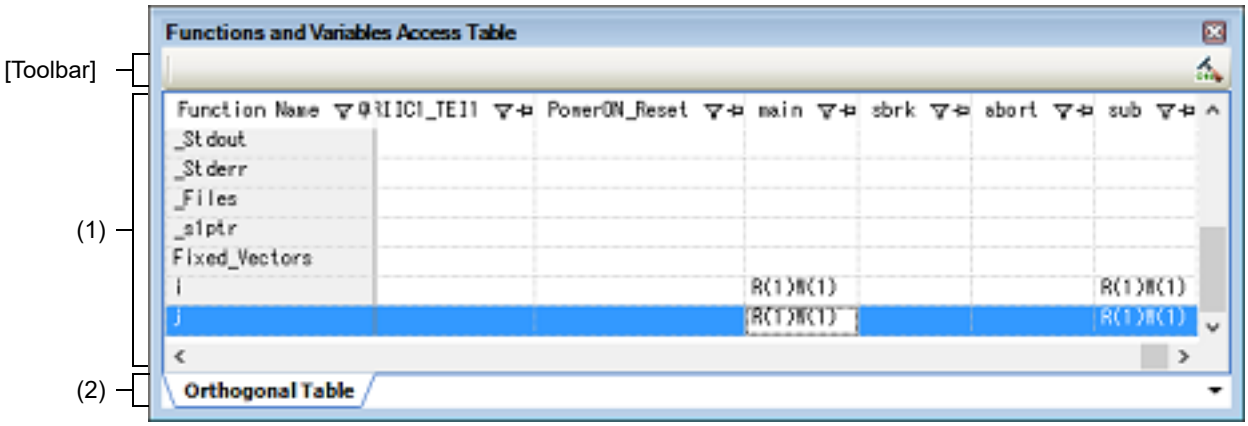
[Function buttons]

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.

Functions and Variables Access Table panel

This panel displays the functions that access variables in the form of an orthogonal table.

Figure A.50 Functions and Variables Access Table Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]

[How to open]

- From the [View] menu, select [Functions and Variables Access Table].

[Description of each area]

(1) Table area

(a) [Orthogonal Table] tab
In the active project, the functions that access variables are displayed in an orthogonal table.
This information is created using the cross reference information and map information. If no information is displayed, click (button for running a build with the build option for generating the orthogonal table made valid) on the right edge of the toolbar and generate information.

The state of functions accessing variables which has been acquired by statically analyzing the C source program is also displayed.

- A cell containing "R" indicates that the value of the variable has been read. The number in parentheses indicates the number of locations at which the variable was read from within the function.
- A cell containing "W" indicates that the value of the variable has been written. The number in parentheses indicates the number of locations at which the variable was written to within the function.

Double-clicking the variable name or cell which shows the state of access to the variable displays that definition in an editor.

(2) Tab selection area
There is the [Orthogonal Table] tab which displays the functions that access variables in the form of an orthogonal table.

[Toolbar]

The toolbar is displayed only when the compiler in use supports generation of cross reference information.

	If the build option related to generation of the cross reference information is invalid, clicking on this button validates the build option and runs a build. If a file is being edited by the text editor at the time, the file is saved.

Measuring Current Consumption panel [E2]

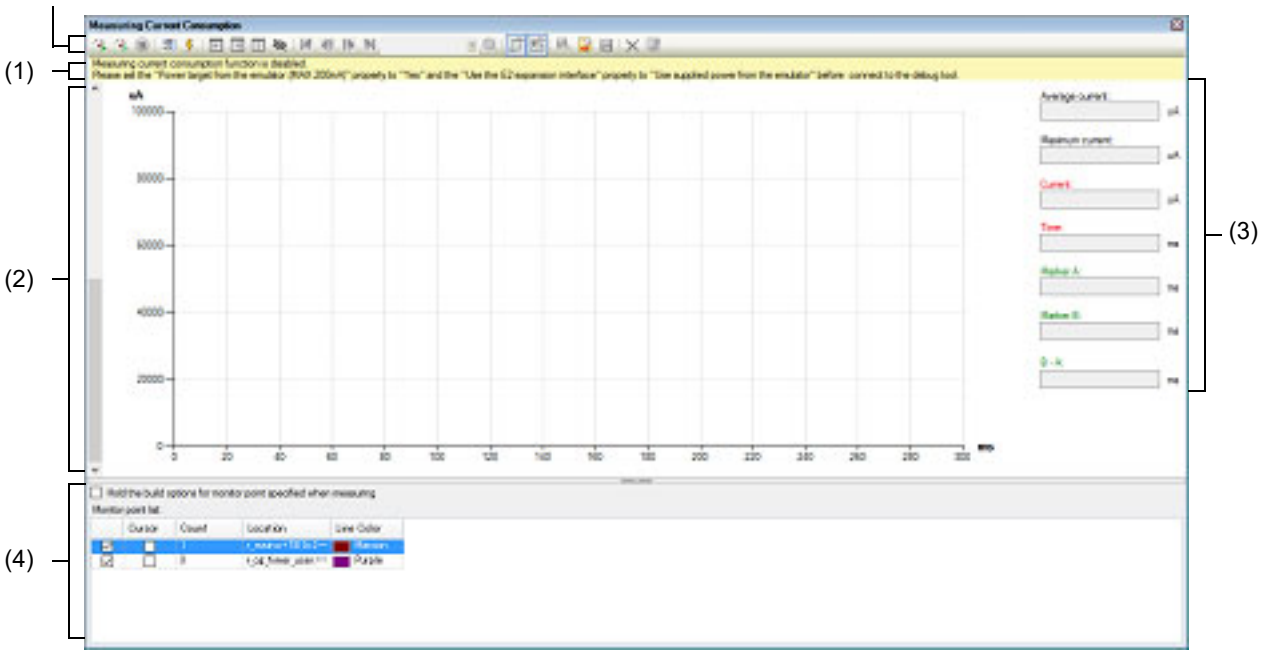
This panel is central to the functionality of the solution for measuring current consumption.
See "2.19 Measuring Current Consumption [E2]" for details on the solution for measuring current consumption.

Caution 1. Measuring current consumption is only available for the RX100 and RX200 Series.

Caution 2. There will be some error in the results of measurement. For the specifications of the measurement of current consumption, e.g. the maximum current consumption, the maximum time over which measurement is possible, the resolution of measurements of current consumption, etc., refer to the E2 Emulator User's Manual.

Figure A.51 Measuring Current Consumption Panel

[Toolbar]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Context menu]

[How to open]

- From the [Debug] menu, select [Debug Solutions] >> [Measuring Current Consumption].
- On the Solution List panel, click the [Go] button of Measuring Current Consumption.

[Description of each area]

(1) Warning display area
This area shows warning messages if there are inconsistencies between the conditions for measurement of current consumption and the conditions to be set before the debug tool is connected.
This area is hidden when there is no warning to be reported.

(2) Graph display area
This area shows the measured results for current consumption with the current represented by the y axis and time by the x axis of the graph.
This area also has the following lines which can be dragged.

Supplementary Line	Description
Cursor	Red vertical line
Marker A, Marker B	Green vertical line
Supplementary lines for searching	Orange horizontal line

When a cursor or marker is dragged, information in the information display area is updated accordingly.
The supplementary lines for searching are used to simply specify the threshold current which is the basis of the search.

Remark While current consumption is being measured, the display of data is cleared and the cursor and markers cannot be dragged.

(3) Information display area
This area displays particular values from the graph of the measured data that is displayed in the graph area.

(a) [Average current]
The average of the values for current between markers A and B is displayed.
When there are no markers, blank space is displayed.

(b) [Maximum current]
The maximum of the values for current between markers A and B is displayed.
When there are no markers, blank space is displayed.

(c) [Current]
The values for current at the position of the cursor is displayed.
When the cursor is not displayed, this item becomes blank.

(d) [Time]
The time at the position of the cursor is displayed.
When the cursor is not displayed, this item becomes blank.

(e) [Marker A]
The time at the position of marker A is displayed.
When marker A is not displayed, this item becomes blank.

(f) [Marker B]
The time at the position of marker B is displayed.
When marker B is not displayed, this item becomes blank.

(g) [B - A]
The time between markers A and B is displayed.
When markers are not displayed, this item becomes blank.

(4) Monitor point list display area
This area displays a list of monitoring points that have been set and is used to specify the settings for building and downloading when monitor points are set.

(a) [Hold the build options for monitor point specified when measuring]
When building proceeds with the function for automatically inserting nop instructions for monitoring points, the options for use in building are reflected in the properties of the build tool.

Caution This check box is only effective when the compiler for the current active project is CC-RX V2.08.00 or later.

(b) [Monitor point list]
A list of monitoring points that are currently set is displayed.
When the measured results are displayed in the graph display area and a cell is double-clicked with each line of the list selected, the cursor in the graph display area is moved to the position of the monitoring point that is indicated by that line.

The following describes the individual components.

<1> Check box
Monitoring points for which the check boxes are selected are valid in the measurement of current consumption.

<2> Cursor
Displays the cursor in the graph display area at the position of the monitoring point on the given line.
When the cursor is selected for a point, the background color of the cell changes to red.

<3> Count
Displays how many times the monitoring point on the given line has been crossed during measurement.

<4> Location
Displays the position of the monitoring point on the given line in the format "*file-name#line-number* (address)".

Example main.c#16(0xFFF0E)

Caution Addresses are only displayed when they have already been determined during measurement.

<5> Line Color
In the graph display area, the color of the vertical line is displayed at the position corresponding to the monitoring point on the given line.

[Toolbar]

	Executes the program and starts measurement of current consumption. When V2.08.00 or later of the CC-RX compiler is selected for the active project, rebuilding and downloading are performed to insert nop instructions at the positions corresponding to monitoring points in the source file registered in the active project.
	Executes the program and starts measurement of current consumption. Rebuilding and downloading are not performed to insert nop instructions at the positions corresponding to monitoring points. Note that this button appears when V2.08.00 or later of the CC-RX compiler is selected for the active project.
	Has the same functionality as the same button on the Debug toolbar in the Main window.
	Opens the Current Consumption Measurement Condition Setting dialog box [E2] to set measurement conditions.
	Opens the Current Consumption Measurement Trigger Condition Setting dialog box [E2], which is used to set trigger conditions for measurement.
	Moves marker A in the graph display area to the corresponding position.
	Moves marker B in the graph display area to the corresponding position.
	Moves the cursor in the graph display area to the corresponding position.
	Switches between showing and hiding the cursor, marker A, marker B, and the supplementary lines for searching in the graph display area.
	Searches for the first position that matches the condition set with the supplementary lines for searching and the search mode. When the position is found, the cursor in the graph display area is moved to that position.
	Searches for the first position before the cursor position that matches the condition set with the supplementary lines for searching and the search mode. When the position is found, the cursor in the graph display area is moved to that position.
	Searches for the first position after the cursor position that matches the condition set with the supplementary lines for searching and the search mode. When the position is found, the cursor in the graph display area is moved to that position.

	Searches for the last position that matches the condition set with the supplementary lines for searching and the search mode. When the position is found, the cursor in the graph display area is moved to that position.
	Selects a search mode from among , , , and . "Under threshold", "In range", "Out of range" or "Monitor point" can be selected as the search mode.
	Opens the Current Consumption Measurement Search dialog box [E2] to specify and search the detailed conditions.
	Selects whether movement of the cursor in the graph display area is linked with the Editor panel. When this icon is selected and the cursor is stopped on a monitoring point, the position of source code where the monitoring point has been set is opened in the Editor panel.
	Selects whether movement of the cursor in the graph display area is linked with the Disassemble panel. When this icon is selected and the cursor is stopped on a monitoring point, the position of source code where the monitoring point has been set is opened in the Disassemble panel. When movement of the cursor is also linked with the Editor panel, linking with the Editor panel is given priority.
	Opens the Save Current Values dialog box [E2] to save the data on current in the measured results of current consumption in a file with the CSV format. The file saved by this button can be displayed in the graph display area by clicking on .
	Reads the CSV file saved by using and displays it in the graph display area.
	Opens the Save File dialog box to save the measured results for current consumption and settings of conditions for measurement in a file with the CSV or Microsoft Office Excel book format.
	Clears the graph showing the measured results in the graph display area.
	Clears the graph of the data which has been read by using in the graph display area.

[Context menu]

Save Results of Current Values	Opens the Save Current Values dialog box [E2] to save the data on current in the measured results of current consumption in a file with the CSV format. The file saved by this menu can be displayed in the graph display area by selecting the [Load Results of Current Values] menu.
Load Results of Current Values	Reads the CSV file saved by using the [Save Results of Current Values] menu and displays it in the graph display area.
Clear Measurement Result	Clears the graph showing the measured results in the graph display area.
Clear Loaded Data	Clears the graph of the data which has been read by using the [Save Results of Current Values] menu in the graph display area.
Time Unit	The following cascade menus are displayed to switch the time unit of the X axis in the graph display area.
Second	Switches the time unit of the X axis in the graph display area to seconds.
Millisecond	Switches the time unit of the X axis in the graph display area to milliseconds.
Microsecond	Switches the time unit of the X axis in the graph display area to microseconds.

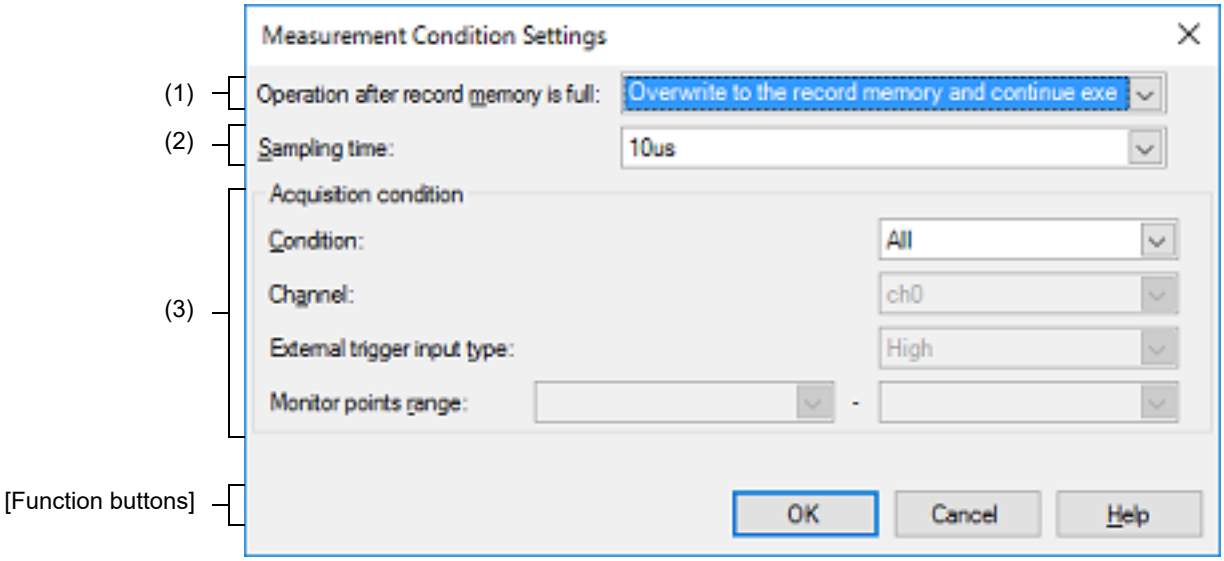
Zoom In	The following cascade menus are displayed to zoom in the display ranges of the X and Y axes in the graph display area.
Zoom In	Zooms in the display ranges of the X and Y axes in the graph display area.
Zoom In X Axis	Zooms in the display ranges of the X axe in the graph display area.
Zoom In Y Axis	Zooms in the display ranges of the Y axe in the graph display area.
Zoom Out	The following cascade menus are displayed to zoom out the display ranges of the X and Y axes in the graph display area.
Zoom Out	Zooms out the display ranges of the X and Y axes in the graph display area.
Zoom Out X Axis	Zooms out the display ranges of the X axe in the graph display area.
Zoom Out Y Axis	Zooms out the display ranges of the Y axe in the graph display area.
Reset Zoom	Restores the display ranges of the X and Y axes in the graph display area to the initial values.

Current Consumption Measurement Condition Setting dialog box [E2]

This dialog box is used to set measurement conditions in the Measuring Current Consumption panel [E2].

Caution Measuring current consumption is only available for the RX100 and RX200 Series.

Figure A.52 Current Consumption Measurement Condition Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Measuring Current Consumption panel [E2], click the button on the toolbar.

[Description of each area]

(1) [Operation after record memory is full]
Select the operation of the debugger after the recording memory of the emulator becomes full during the measurement of current consumption from the following drop-down list.

Overwrite to the record memory and continue execution	Continue to execute the program and to record data by overwriting the data from the beginning of the recording memory.
Stop recording	Only stop writing to the recording memory but continue to execute the program.
Stop program	Stop both writing to the recording memory and execution of the program.

(2) [Sampling time]
Select the sampling time during the measurement of current consumption.
The following sampling times can be selected.
1us, 2us, 5us, 10us, 20us, 50us, 100us, 200us, 500us, 1ms

(3) [Acquisition condition] area
Make settings regarding conditions for writing to the recording memory of the emulator.

(a) [Condition]
Select the type of condition for writing to the recording memory of the emulator from the following drop-down list.

All	Record all measured data without restrictions.
During detecting an external trigger input	Record the measured data with use of the function for detecting an external trigger input in the extended interface of the E2 emulator only while the external trigger input is detected.
During monitor points	Record the measured data only over the time from detection of the monitoring point for starting measurement until detection of the monitoring point for ending measurement.

(b) [Channel]
When "During detecting an external trigger input" is selected for [Condition], select "ch0" or "ch1" as the external trigger input channel from the extended interface of the E2 emulator that is to be detected.

(c) [External trigger input type]
When "During detecting an external trigger input" is selected for [Condition], select "High" or "Low" as the level of the external trigger input signal to be detected.

(d) [Monitor points range]
When "During monitor points" is selected for [Condition], select the two target monitoring points.
Monitoring points that have currently been specified for the project are displayed in a drop-down list.
The measured data will only be recorded from the time the monitoring point set in the left side has been detected until detection of the monitoring point set on the right side.

[Function buttons]

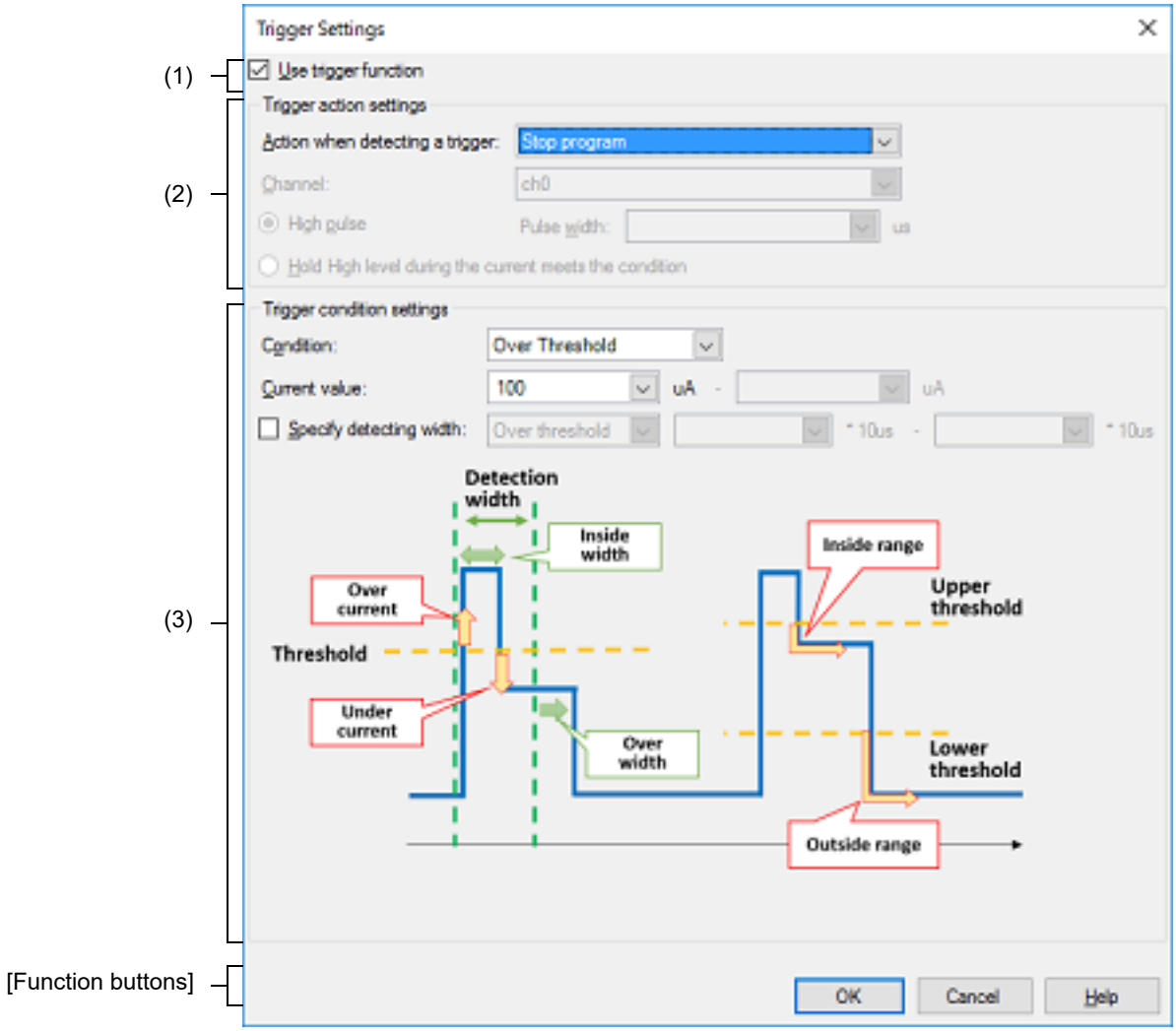
Button	Function
OK	Reflects the settings of the measurement conditions that were made in this dialog box to the Measuring Current Consumption panel [E2].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Current Consumption Measurement Trigger Condition Setting dialog box [E2]

This dialog box is used to set trigger conditions in the Measuring Current Consumption panel [E2].

Caution Measuring current consumption is only available for the RX100 and RX200 Series.

Figure A.53 Current Consumption Measurement Trigger Condition Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Measuring Current Consumption panel [E2], click the button on the toolbar.

[Description of each area]

(1) [Use trigger function]
Specify whether to use the trigger function.

(2) [Trigger action settings] area
Set the operation of the E2 emulator on detection of the trigger.

(a) [Action when detecting a trigger]
Select the operation on detection of the trigger from the following drop-down list.

Stop program	Stop execution of the program.
Output external trigger signal	Output the external trigger signal by using the function for output of an external trigger signal from the extended interface of the E2 emulator.

(b) [Channel]
When "Output external trigger signal" is selected for [Action when detecting a trigger], select "ch0" or "ch1" as the external trigger output channel of the extended interface of the E2 emulator.

(c) External trigger output type
Select [High pulse] or [Hold High level during the current meets the condition] as the type of signal for output as the external trigger.
When [High pulse] is selected, specify the width of the pulse as a decimal value between 1 and 65535 (in microseconds).

(3) [Trigger condition settings] area
Set the trigger conditions.

(a) [Condition]
Select the type of change in values for current as the trigger condition from the following drop-down list.

Over Threshold	Detect the moment when the current being measured becomes or exceeds a specified value.
Under Threshold	Detect the moment when the current being measured becomes or falls below a specified value.
In Range	Detect the moment when the current being measured enters a specified range of values.
Out of Range	Detect the moment when the current being measured leaves a specified range of values.

(b) [Current value]
Specify the specific value or range of values for current to serve as the trigger condition in microamperes.
Specify one value for current when "Over threshold" or "Under Threshold" is selected for [Condition], and two values for current when "In range" or "Out of Range" is selected for [Condition].

Caution For a range of current values, the value entered in the left field for entry must be lower than that in the right field.

(c) [Specify detecting width]
When "Over threshold", "Under Threshold", or "In range" is selected for [Condition], specify whether the trigger is only detected when the change in the value for current continues for a specific period or at the instant the required change in the value is detected.
To select detection of the trigger, select the check box and specify the condition for the interval and the interval.

<1> Condition for the interval
Select the condition for the interval from the following drop-down list.

Over threshold	Detect the trigger when the condition specified for [Condition] continues to be satisfied for at least the specified interval.
In range	Detect the trigger only when the condition specified for [Condition] continues to be satisfied within the specified interval.

<2> Interval
Specify a multiple of the sampling time selected for [Sampling time] in the Current Consumption Measurement Condition Setting dialog box [E2].
The actual interval is the specified multiple of the sampling time.

[Function buttons]

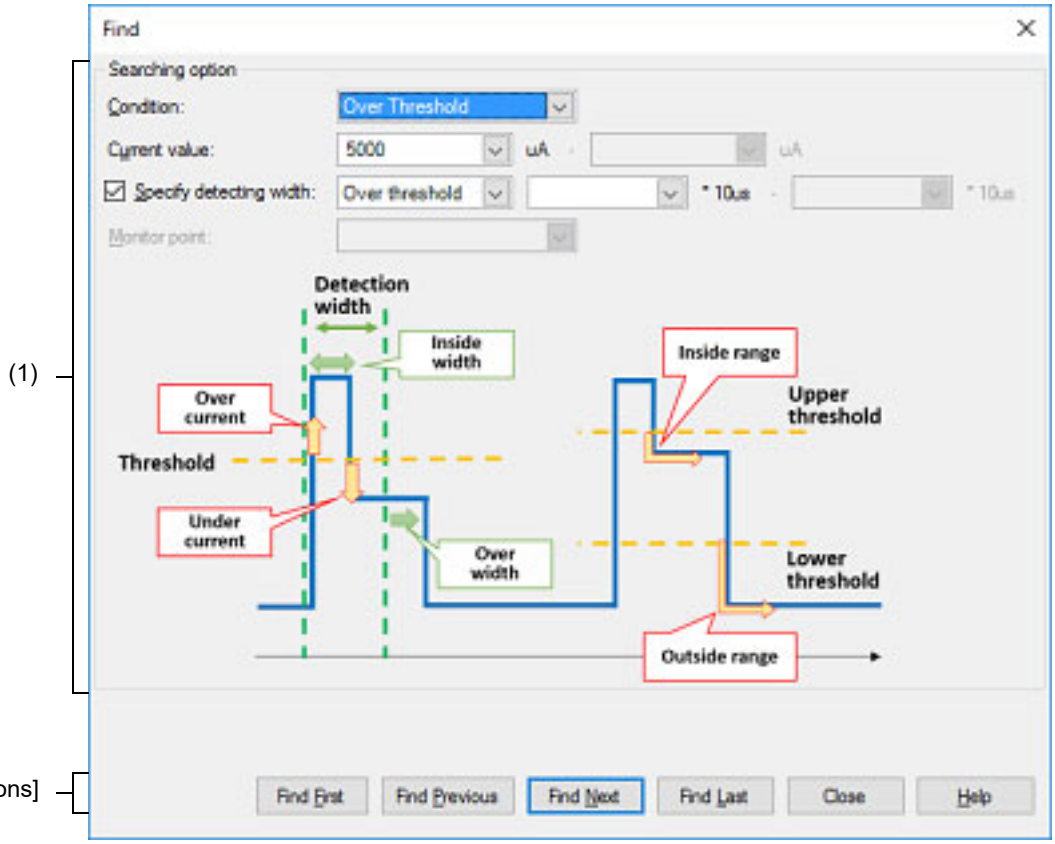
Button	Function
OK	Reflects the settings of the trigger conditions that were made in this dialog box to the Measuring Current Consumption panel [E2].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Current Consumption Measurement Search dialog box [E2]

This dialog box is used to specify and search the detailed conditions in the Measuring Current Consumption panel [E2].

Caution Measuring current consumption is only available for the RX100 and RX200 Series.

Figure A.54 Current Consumption Measurement Search Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Measuring Current Consumption panel [E2], click the button on the toolbar.

[Description of each area]

(1) [Searching option] area
Specify the search conditions.

(a) [Condition]
Select the type of the search conditions from the following drop-down list.

Over threshold	Detect the moment when the current being measured becomes or exceeds a specified value.
Under Threshold	Detect the moment when the current being measured becomes or falls below a specified value.

In range	Detect the moment when the current being measured enters a specified range of values.
Out of Range	Detect the moment when the current being measured leaves a specified range of values.
Monitor point	Detect the specified monitoring point.

(b) [Current value]
Specify the specific value or range of values for current to serve as the search condition in microamperes. Specify one value for current when "Over threshold" or "Under Threshold" is selected for [Condition], and two values for current when "In range" or "Out of Range" is selected for [Condition].

Caution For a range of current values, the value entered in the left field for entry must be lower than that in the right field.

(c) [Specify detecting width]
When "Over threshold", "Under Threshold", or "In range" is selected for [Condition], specify whether the data is only searched when the change in the value for current continues for a specific period or at the instant the required change in the value is detected.
To search the data, select the check box and specify the condition for the interval and the interval.

<1> Condition for the interval
Select the condition for the interval from the following drop-down list.

Over threshold	Detect the data when the condition specified for [Condition] continues to be satisfied for at least the specified interval.
In range	Detect the data only when the condition specified for [Condition] continues to be satisfied within the specified interval.

<2> Interval
Specify a multiple of the sampling time selected for [Sampling time] in the Current Consumption Measurement Condition Setting dialog box [E2].
The actual interval is the specified multiple of the sampling time.

(d) [Monitor point]
When "Monitor point" is selected for [Condition], select "All" or the current monitoring point as the target monitoring point.

[Function buttons]

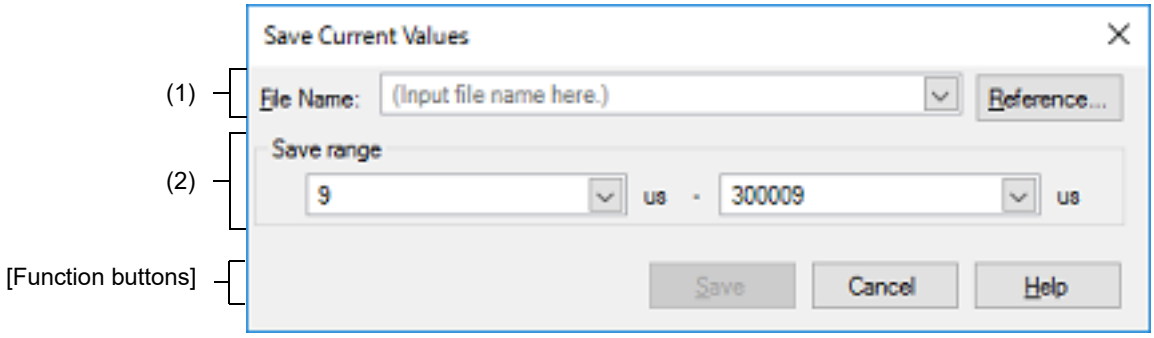
Button	Function
Find First	Searches the whole of the measured data from its start for points that satisfy the conditions specified in this dialog box.
Find Previous	Searches forward in time from the position of the cursor in the graph display area in the Measuring Current Consumption panel [E2] for the points that satisfy the conditions specified in this dialog box.
Find Next	Searches backward in time from the position of the cursor in the graph display area in the Measuring Current Consumption panel [E2] for the points that satisfy the conditions specified in this dialog box.
Find Last	Searches the whole of the measured data from its end for points that satisfy the conditions specified in this dialog box.
Close	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Save Current Values dialog box [E2]

This dialog box is used to save the values for current in the Measuring Current Consumption panel [E2].

Caution Measuring current consumption is only available for the RX100 and RX200 Series.

Figure A.55 Save Current Values Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Measuring Current Consumption panel [E2], click the button on the toolbar.

[Description of each area]

(1) [File Name] area
Specify the name of the file to save.
You can either type a filename directly into the text box (up to 259 characters), or select one from the input history via the drop-down list (up to 10 items).
You can also specify the file by clicking the [Reference…] button, and selecting a file via the Select Current Values Data Save File dialog box.
When only the name of the file is specified, i.e. path information is not included, the file will be in the project folder.

(2) [Save range] area
Specify the range in time of values for current to be saved by specifying the times (in microseconds) where saving is to start and end. The numbers are decimal.
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).
When "All data" is selected, the values for current from the minimum time as the start time to the maximum time as the end time will be saved.

[Function buttons]

Button	Function
Save	Saves the data to a file with the specified filename, in the CSV format.
Cancel	Cancels the save and closes this dialog box.
Help	Displays the help for this dialog box.

B. I/O FUNCTIONS

This section describes the I/O 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
The I/O function for a simulator can be used when [Simulator mode] is specified in the [Select stream I/O mode] property in the [Stream I/O] [Simulator] category on the Property panel's [Debug Tool Settings] tab.

B.1 Standard I/O and File I/O

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 I/O.
With the standard I/O 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 I/O functions are listed below.

Table B.1 I/O Facilities

Classification	Facility name	Description
Standard I/O	GETC	Accepts 1 byte from standard input.
	PUTC	Outputs 1 byte to standard output.
File I/O	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 "CC-RX Compiler User's Manual".

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 I/O 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 I/O 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 I/O function.

MSB LSB

- Reservation of areas for the parameter block and stream I/O 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	*Beginning address of output buffer*	
+2		

[Parameters]

Parameter	Description
Beginning address of output buffer (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	*Execution result*	*File number*
+2	*Open mode*	Unused
+4	*Beginning address of file name*	
+6		

[Parameters]

Parameter	Description
Execution result (output)	0: Terminated normally -1: Error
File number (output)	Number that is used for access to files after the open process
Open mode (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.
Beginning address of file name (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	*Execution result*	*File number*

[Parameters]

Parameter	Description
Execution result (output)	0: Terminated normally -1: Error
File number (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	*Execution result*	*File number*
+2	Unused	
+4	*Beginning address of input buffer*	
+6		

[Parameters]

Parameter	Description
Execution result (output)	0: Terminated normally -1: EOF detected
File number (input)	Number that is returned when a file is opened
Beginning address of input buffer (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	*Execution result*	*File number*
+2	Unused	
+4	*Beginning address of output buffer*	
+6		

[Parameters]

Parameter	Description
Execution result (output)	0: Terminated normally -1: Error
File number (input)	Number that is returned when a file is opened
Beginning address of output buffer (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)]

Example 1.

	1 byte	1 byte
+0	*Execution result*	*File number*

[Parameters]

Parameter	Description
Execution result (output)	0: Not EOF -1: EOF detected
File number (input)	Number that is returned when a file is opened

[Detailed Description]

- Checks for end of file.

[Example]

```
_fpeof:
        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	*Execution result*	*File number*
+2	*Direction*	Unused
+4	*Offset*	
+6		

[Parameters]

Parameter	Description
Execution result (output)	0: Terminated normally -1: Error
File number (input)	Number that is returned when a file is opened
Direction (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
Offset (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	*Execution result*	*File number*
+2	Unused	
+4		
+6	*Offset*	

[Parameters]

Parameter	Description
Execution result (output)	0: Terminated normally -1: Error
File number (input)	Number that is returned when a file is opened
Offset (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]

```
_fptell:
        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
```

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Jun 01, 2021	-	First Edition issued

CS+ V8.06.00 User's Manual:
RX Debug Tool

Publication Date: Rev.1.00 Jun 01, 2021
Published by: Renesas Electronics Corporation

CS+ V8.06.00

Renesas Electronics Corporation

R20UT4977EJ0100