User's Manual

RENESAS

CS+ V8.13.00

Integrated Development Environment

User's Manual: RH850 Debug Tool

Target Device

RH850 Family

Renesas Electronics
www.renesas.com

Rev.1.00 2024.11

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.

How to Use This Manual

This manual describes the role of the CS+ integrated development environment for developing applications and systems for RH850 family, and provides an outline of its features.

CS+ is an integrated development environment (IDE) for RH850 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
How to Read This Manual	It is assumed that the readers of this manual have general knowledge of electricity, logic circuits, and microcontrollers.

Conventions

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

TABLE OF CONTENTS

1. GENERAL

CS+ is a platform of an integrated developing environment for RH850 family, RX family, V850 family, RL78 family, 78K0R microcontrollers, 78K0 microcontrollers.
CS+ can run all the operations needed for developing the programs such as designing, cording, building, debugging, and flash programming.
In this manual, the debugging is explained out of those operations needed for the program development.

Caution 1. When the E2 emulator (abbreviated name: E2) is used, please read references to "E1" in this manual as also meaning "E2".

Caution 2. When the IE850A is used, please read references to "E1" in this manual as also meaning "IE850A".

In this chapter, an overview of debugging products that CS+ provides is explained.

1.1 Summary

You can effectively debug/simulate the program developed for the RH850 family, using the debugger which CS+ provides.

1.2 Features

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

- Execution and breaks in execution for multi-core devices
  When the microcontroller is a type that has multiple cores, synchronous or asynchronous execution and synchronous or asynchronous breaks are selectable.
  A core in the initial stop state or in standby mode can also be debugged.
  Switching the selection of the core leads to the display of information on the selected core in the panels.

- Debugging the virtualization facility
  When a microcontroller incorporates a virtualization facility, debugging can be focused on the virtual machines rather than on the microcontroller as a whole.

- Connecting to the various debug tools
  A pleasant debugging environment for target systems is provided by connecting to the full-spec emulator (Full-spec emulator), the on-chip debugging emulator (E1/E20) and Simulator.

- C source text and disassembled text are shown mixed
  The C source text and the disassembled text are shown mixed on the same panel.

- Source level debugging and instruction level debugging
  The source level debugging and the instruction level debugging for a C source program can be done.

- Real-time display update function
  The contents of memory, registers and variables are automatically updated not only when the program execution is stopped, but also in execution.

- Save/restore 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 a debugging process of CS+ and main functions for debugging.

2.1 Overview

The basic debugging sequence for programs using CS+ is as follows:

(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 an existing one.

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

(3) Create a load module
Create a load module by running a build after setting of the active project and the build tool to be used.

Remark For details on "Create a load module" with CC-RH, see "CS+ Integrated Development Environment User's Manual: RH850 Build".

(4) Confirm the connection to a host machine
Connect the debug tool (Full-spec emulator, E1, E20, or Simulator) to be used to a host machine.

(5) Select the debug tool to use
Select the debug tool to be used in a project.

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

(6) Configure the operating environment of the debug tool
Configure the operating environment of the debug tool selected in steps (5).

- [Full-spec emulator]/[IE850A]/[E1]/[E20]/[E2]
- [Simulator]

(7) Connect to the debug tool
Connect the debug tool to CS+ to start communication.

(8) Execute downloading
Download the load module created in steps (3) to the debug tool.

(9) Display 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 by using the operation method corresponding to a purpose.
If you wish to stop the program at the arbitrary position, set a breakpoint/break event[Note] before executing the program (see "2.11.3 Stop the program at the arbitrary position (breakpoint)", "2.11.4 Stop the program at the arbitrary position (break event)", or "2.11.5 Stop the program with the access to variables/I/O registers").

Note These functions are implemented by setting events to the debug tool used.
See "2.19.7 Notes for setting events", when you use events.

Remark When the selected microcontroller version supports multi-core, select a core (PE: Processer Element) to be debugged before executing the program (see "2.9 Select a Core (PE)").

(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 set break condition is met.

(12) Check the result of the program execution
Check the following information that the debug tool acquired by the program execution.

- Display/Change the Memory, Register and Variable
- Display Information on Function Call from Stack

- Collect Execution History of Programs[Note]
- Measure Execution Time of Programs[Note]
- Measure Performance [Full-spec emulator][E1][E20][Note]
- Measure Coverage [Simulator]

Note These functions are implemented by setting events to the debug tool used.
See "2.19.7 Notes for setting events", 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
- Use Hook Function

Remark 2. The acquired information can be saved to a file.

- Save the disassembled text contents
- Save the memory contents
- Save the CPU register contents
- Save the I/O register contents
- Save the contents of local variables
- Save the contents of watch-expressions
- Save the contents of call stack information
- Save the contents of execution history

(13) Execute uploading
Save the program (the memory contents) to a file in the arbitrary format (e.g. Intel HEX file, Motorola S-record file, binary file, and etc.), as required.

(14) Disconnect from the debug tool
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

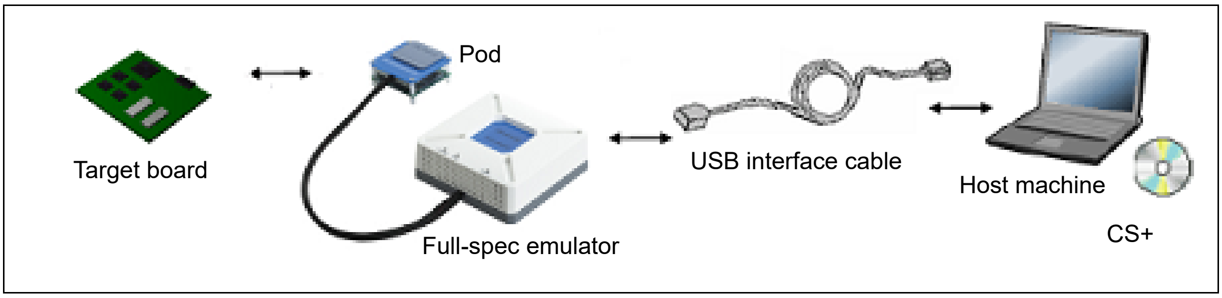
Connection examples for each debug tool are shown.

2.2.1.1 [Full-spec emulator]
2.2.1.2 [E1]
2.2.1.3 [E20]
2.2.1.4 [Simulator]

2.2.1.1 [Full-spec emulator]

Connect a host machine and Full-spec emulator. If required, connect a target board, too.
For details on the connection method, see the user's manual for Full-spec emulator.

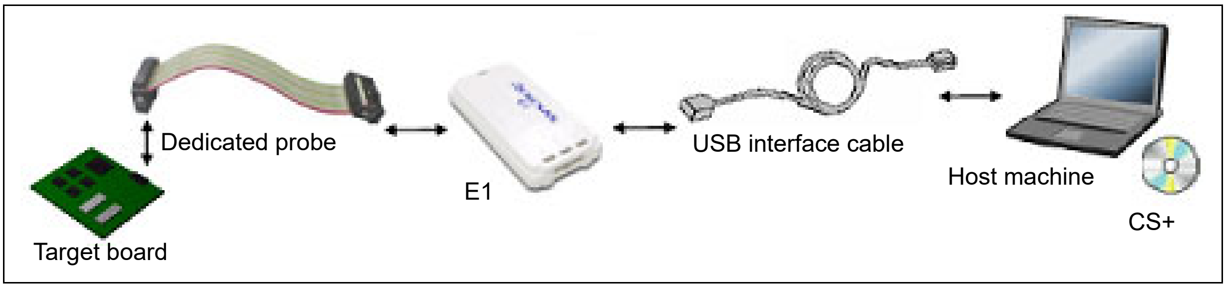
Figure 2.1 Connection Example [Full-spec emulator]



2.2.1.2 [E1]

Connect a host machine and E1. If required, connect a target board, too.
For details on the connection method, see the user's manual for E1.

Figure 2.2 Connection Example [E1]



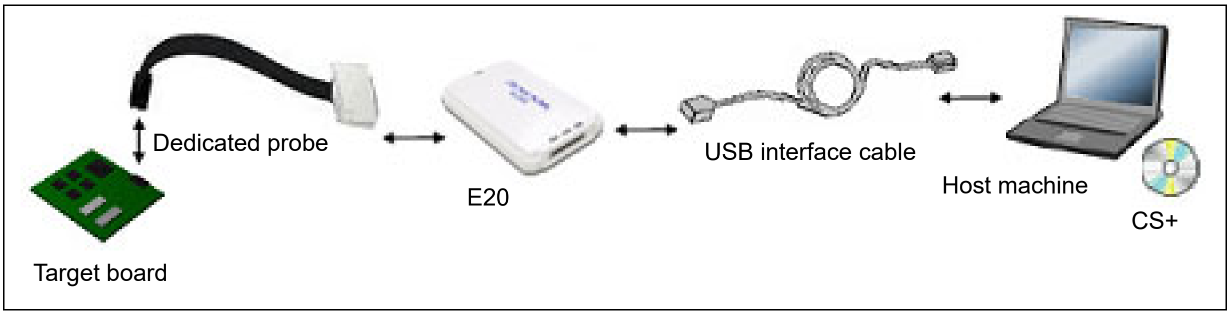
Caution 1. Only the Low Pin Debug interface (hereafter referred to as LPD communications) is supported for communication with the target board.

Caution 2. For details on the connection using a debug MCU board, see the user's manual for debug MCU board.

2.2.1.3 [E20]

Connect a host machine and E20. If required, connect a target board, too.
For details on the connection method, see the user's manual for E20.

Figure 2.3 Connection Example [E20]



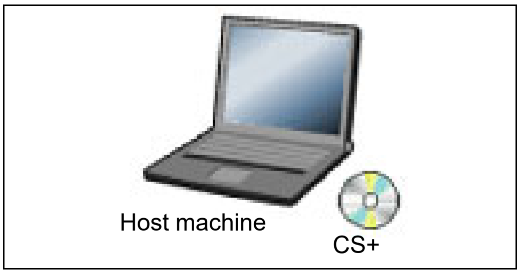
Caution 1. Only the Low Pin Debug interface (hereafter referred to as LPD communications) is supported for communication with the target board.

Caution 2. For details on the connection using a debug MCU board, see the user's manual for debug MCU board.

2.2.1.4 [Simulator]

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

Figure 2.4 Connection Example [Simulator]



2.3 Configuration of Operating Environment of the Debug Tool

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

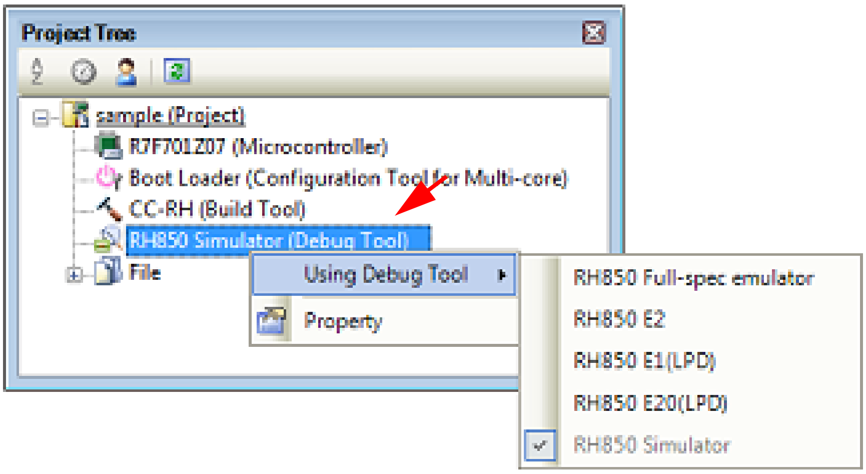
2.3.1 Select the debug tool to use

You can configure the operating environment in the Property panel corresponding to the debug tool to use.

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

To select or switch the debug tool, use the context menu shown by right clicking on the [RH850 *Debug tool name* (Debug Tool)] node on the Project Tree panel.

Figure 2.5 Select/Switch Debug Tool to Use



Caution The context menu items displayed differ depending on the microcontroller selected in the project.

If the Property panel is already open, click the [RH850 *Debug tool name* (Debug Tool)] node again. The view switches to the Property panel of the selected debug tool.

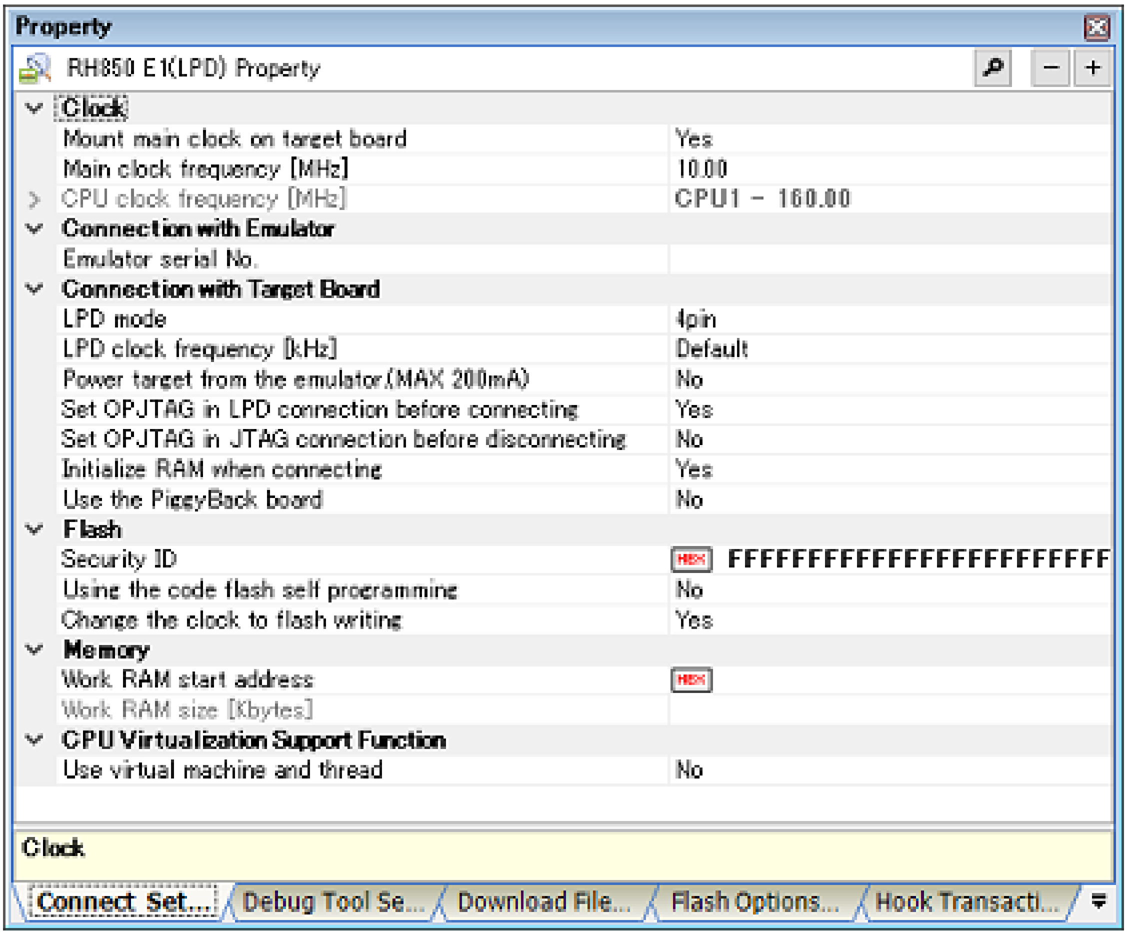
If the Property panel is not open, double-click the above mentioned node to open the corresponding Property panel.

2.3.2 [Full-spec emulator]/[IE850A]/[E1]/[E20]/[E2]

Configure the operating environment on the Property panel below when using Full-spec emulator, IE850A, E1, E20, or E2.

Caution Only LPD communications are supported for communication with the target board.

Figure 2.6 Example of Property Panel



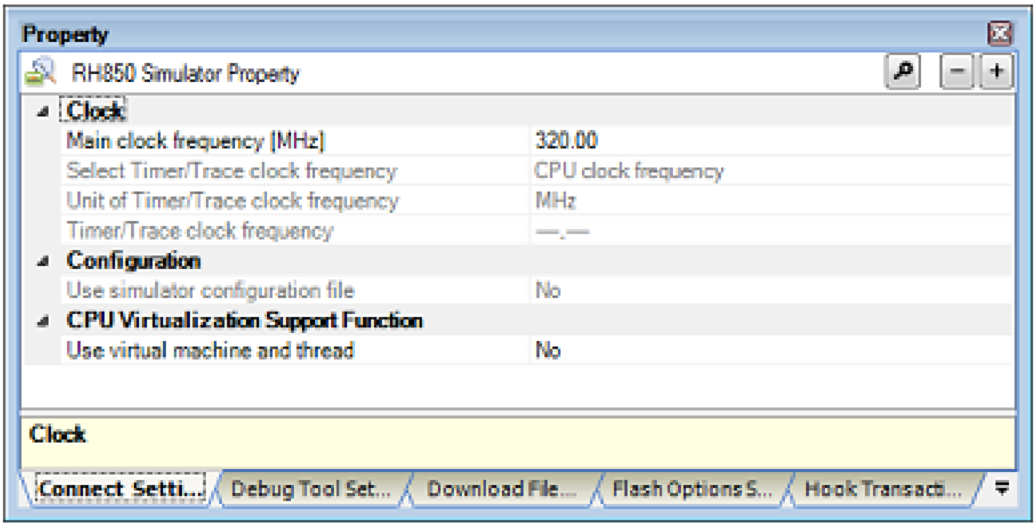
Follow the steps below by selecting the corresponding tab on the Property panel.

[Connect Settings] tab
[Debug Tool Settings] tab
[Download File Settings] tab
[Flash Options Settings] tab
[Hook Transaction Settings] tab

2.3.3 [Simulator]

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

Figure 2.7 Example of Property Panel [Simulator]



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

[Connect Settings] tab
[Debug Tool Settings] tab
[Download File Settings] tab
[Flash Options Settings] tab
[Hook Transaction Settings] tab

The simulator is used to simulate instructions of the CPU core assuming that each CPU core type of target MCUs has the number of MPU areas and does or does not have an instruction cache as listed below.

Table 2.1 Number of MPU Areas and Presence of an Instruction Cache per CPU Core Type

CPU Core Type	Number of MPU Areas	Instruction Cache (with Size and Number of Ways)
RH850G3M	12	Yes (8 Kbytes, 4 Way)
RH850G3K	8	No
RH850G3MH	16	Yes (8 Kbytes, 4 Way)
RH850G3KH	16	No
RH850G4MH, RH850G4KH	Depends on the specifications of the target MCU.	

The following notes are also applicable.

(1) CPU operation clock
The CPU clock operates on the frequency set up with the [Main clock frequency [MHz]] property on the [Connect Settings] tab.

(2) Access latency
Since the access latency for various memories and peripheral modules is not considered, the execution time (count of cycles) is different from the target device.
Therefore, the following results of the simulator are different from the target device.

- Measurement result by Run-Break Timer
- Measurement result by Timer Result event
- [Pipeline] area in the Trace panel

- [Timer] area in the Trace panel
- Trace result when selecting [All core] in the [Trace target] property on the [Debug Tool Settings] tab (Timing between PEs)
- Time stamp of software trace data

The above sets of information are used as dynamic analysis information for the analysis tools but will differ from results for the actual target devices.

(3) Peripheral function
The simulator doesn't support simulation of peripheral functions.
The following Python functions can be used to generate pseudo interrupts.
See "CS+ Integrated Development Environment User's Manual: Python Console" for detail.

- debugger.Interrupt.OccurEI
- debugger.Interrupt.OccurFE
- debugger.Interrupt.RequestEI
- debugger.Interrupt.RequestFE
- debugger.Interrupt.RequestFENMI

(4) Supported memory
The simulator supports the following types of memory.
Note that with the simulator, however, areas that require special procedures for access according to the specifications of the devices are accessible without following the procedures.

[RH850G3M, RH850G3K, RH850G3MH, RH850G3KH]
External Memory/APB/Data Flash/PBUS/HBUS/Local RAM/Global RAM/AXI/Retention RAM/CPU Peripheral/Code Flash/Video RAM/SDRAM

Remark Note that access to Retention RAM, Video RAM, and SDRAM is handled in the same way as access to Global RAM.

[RH850G4MH, RH850G4KH]
External Memory/APB/Data Flash/PBUS/HBUS/Local RAM/Cluster RAM/Retention RAM/IBUS/CPU Peripheral/Code Flash/Security Setting Area/Configuration Setting Area/Block Protection Area/Extended Data Area/GTM RAM/Erase Counter/Switch Area/Tag Area/DFP RAM/DFP Peripheral

Remark 1. The Mirror Area and Blank check Area in the Code Flash memory are not supported.

Remark 2. Access to the Security Setting Area, Configuration Setting Area, Block Protection Area, Extended Data Area, GTM RAM, Erase Counter, Switch Area, Tag Area, DFP RAM and DFP Peripheral is handled in the same way as access to the External Memory.

(5) System register
System registers that depend on the hardware specifications of the given product, such as the control function registers for the LSU and data buffer and the hardware function registers are not supported.

(6) Option bytes
The simulator doesn't support the option bytes.

(7) Differences between operation of the target device and the simulator

- Since the timing of instruction fetching differs between the target device and the simulator, the instruction cache or the number of rounds of access will differ from that on the actual device.
- Even if the target device requires appropriate synchronization processing such as dummy reading or SYNCI instructions, the simulator may operate without such processing.
- Although the cache hit rate (the rate of the number of hits to the number of rounds of cache access) is displayed on the [Cache rate] tab in the Output panel, the rate may differ from that on the target device.
- The error-correcting code (ECC), LRU operation, and information on cache errors in the instruction cache are not supported.
- For FPU instructions and FXU instructions (only for the RH850G4MH), the results of operations will differ between the target device and the simulator.
- The values of PID may differ from those on the target devices.

- [RH850G3K]
  On the target device, the same processing is handled for the LDL.W/STC.W and LD.W/ST.W instructions. On the simulator, the LDL.W/STC.W instruction operates in the same way as on the RH850G3M.
- [RH850G4MH, RH850G4KH]
  Values counted by the count function system registers (performance counter and timestamp counter) are in error relative to those for the target device, so these values do not match.
- Some RH850 MCUs have specific option bytes for selecting the boot mode for the CPUs and IO registers for selecting which CPUs should be booted up when the MCU is reset. The simulator, on the other hand, does not support such facilities and all CPUs are booted up when the MCU is reset.

- Copyright of the RH850G4MH, RH850G4KH simulator
  The use of SoftFloat technology by this software is under the License for Berkeley SoftFloat Release 3e.
  Copyrights of other software components are owned by Renesas Electronics Corporation.

2.4 Applicable Debugging Methods

This section describes debugging methods which are applicable to debugging specific facilities of RH850 family devices.

2.4.1 Debugging the Initial Stop State or Standby Mode

This section describes how to enable the debugging facilities for the initial stop state or standby mode and gives notes on usage when the selected microcontroller is in the initial stop state or incorporates a specific standby-mode facility.

Caution The simulator does not support this facility.

In some descriptions, this section refers to the states of the microcontroller listed below as well as those in the Main window.

Table 2.2 States of the Microcontroller

Name	State of the Microcontroller
Stop	In the stop mode
Initial Stop	In the initial stop state
Deep Stop	In the deep stop mode
Cyclic Run	In the cyclic run mode
Cyclic Stop	In the cyclic stop mode
Cyclic Disable	The state of cores other than the main core when the main core is in the cyclic run mode or cyclic stop mode

- How to enable the debugging facilities
  Set the following for debugging the initial stop state or standby mode.
  The [Connection with Target Board] category on the [Connect Settings] tab in the Property panel

Debug the initial stop state and the standby mode	Yes

- Notes on usage
  (a) If this facility is not used, a program that includes a transition to standby mode cannot be debugged.
  (b) If this facility is used, the following facilities are not available.
    - Asynchronous debugging mode[Note]
    - LPD output of software tracing
    - External trigger input
    - External trigger output
    - Measuring CAN bus reception processing times

  Note In asynchronous debugging mode, the following method is available but is restricted to the initial stop state.
  Execute after resetting microcontroller (CPU) (Initial stop debug)

  (c) Using this facility enables debugging of a core (PE) in the following states.
    - Initial Stop
    - Cyclic Run
    - Cyclic disable in the cyclic run mode

In the cyclic run mode, however, the following facilities that involve the programming of flash memory cannot be used.

- Downloading to flash memory
- Setting and deleting software breaks

When a core is in the following states, executing a program, referring to the CPU registers, and setting and deleting breakpoints are not possible.

- Initial Stop
- Cyclic disable in the cyclic run mode

When the core is in the following states, debugging is not possible. Only a subset of the debugging facilities, such as referring to states, is available.

- Stop
- Deep Stop
- Cyclic Stop
- Cyclic disable in the cyclic stop mode

2.4.2 Debugging the Virtualization Facility

This section describes the debugging facilities for a virtualization facility when a selected microcontroller incorporates that facility.
When specifying contexts for debugging in CS+, debugging with the focus on specific virtual machines (referred to as guests) is enabled.

Select the contexts for debugging in the Select Contexts on Debug target dialog box. The following describes the behavior of the facilities of CS+ when contexts for debugging are selected. The operation of each facility depends on the core (PE) selected as the target for debugging.

(1) Execution and step execution of programs
The operation of a program varies with the setting for the [Skip contexts on not debug target] property in the [Hardware-assisted Virtualization] category on the [Debug Tool Settings] tab page of the Property panel

- When [Yes] is selected
If any break source condition is satisfied and stops the execution of the program, the program is still stopped if the current context has been selected as a target for debugging. If the program was stopped in a context that is not selected as a target for debugging, execution of the program is automatically continued until it makes a transition to a context for debugging and execution only stops after the transition is completed.
- When [No] is selected
If the execution of the program is stopped, the program will be stopped regardless of the context.

Selecting [Yes] enables confirming the operation of the program with the focus only on the contexts for debugging.

(2) Generation of events
For hardware break events, trace events, point trace events, timer measurement events, and performance measurement events, only those generated within the contexts for debugging are detected.

Caution Software break events are detected regardless of the context.

(3) Display of memory and I/O registers
The whole memory and all I/O registers are available for display regardless of the contexts for debugging.

(4) Display of the CPU registers
Register values in the context where execution was stopped are displayed.
However, when the program is stopped in a context other than one that is a target for debugging, [?] is displayed except for the following registers.
PC, R3 (SP), R28, and R31 (LP)

(5) Collection of the history of execution
Only the history of execution in the contexts for debugging is collected.
The timestamps, however, include the execution times of contexts other than those for debugging.
Information on the context is displayed as part of the trace data in the Trace panel.

[IE850A][E2]
The following setting is required to show information on the context.
Select [Yes] for the [Trace the transition information of CPU operation mode] property in the [Trace] category on the [Debug Tool Settings] tab page of the Property panel.

(6) Measurement of execution times
For run-break timer events, times in all contexts are measured regardless of the context for debugging.
For timer measurement events, times in all contexts are measured regardless of the context for debugging when measurement proceeds after an event condition has been satisfied.

(7) Measurement of performance [IE850A] [E2]
Performance is only measured in the contexts for debugging.

(8) Measurement of coverage [Simulator]
All contexts are measured regardless of the context for debugging.

2.4.3 Debugging the GTM

This section describes the methods to enable the debugging facilities for the generic timer IP module (GTM) from Robert Bosch GmbH, operation of each facility, and notes on usage if the selected microcontroller incorporates the GTM.

Caution The simulator does not support debugging of this module.

In some statements, the states of the GTM are referred to in a similar way as in the Main window as follows.

Table 2.3 States of the Microcontroller

Displayed Item	State of the GTM
Standby	A clock signal is not being supplied to the GTM.
Disable	The selected multi-channel sequencer (MCS) is not activated.

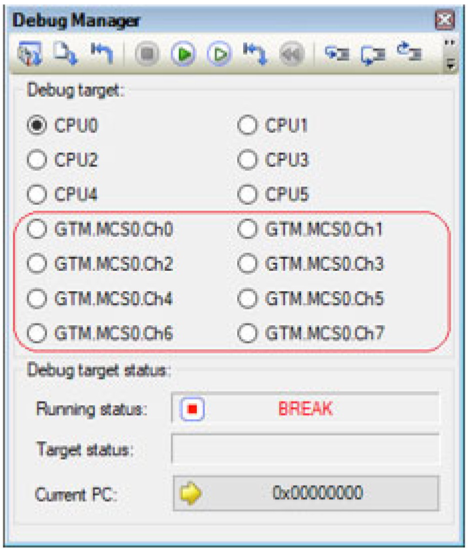
- Enabling the debugging facilities
When the GTM is to be debugged, make the following settings.
[Connection with Target Board] category on the [Connect Settings] tab in the Property panel.

[Debug the GTM function] property	Yes
[MCS to be debug] property	*MCS* to be debugged

With the above settings, it is possible to select a channel of the MCS which is specified for [MCS to be debugged] in the Statusbar of the Main window or in the Debug Manager panel, as the targets for debugging.

Example When "MCS0" is specified for [MCS to be debugged], the channels of that MCS are displayed as the targets for debugging in the Debug Manager panel.

Figure 2.8 Example of the Display in the Debug Manager Panel



- Operation of individual facilities

When the GTM is selected as the target for debugging, each facility behaves as described below.

(1) Execution of a program
Synchronous execution and synchronous breaks are available for all channels of the selected MCS.

(2) Events
Valid events are specifiable for all channels of the MCS.

(3) Memory
The RAM of the MCS is accessible. The displayed addresses are those in the memory space of the MCS which is selected for the [MCS to be debugged] property.

(4) I/O registers
The I/O registers in the GTM are accessible.
The displayed addresses are those in the memory space of the GTM.

(5) CPU registers
The CPU registers for each channel of the MCS are accessible.

(6) Collect Execution History of Programs
The trace data of the channels of the MCS selected in the [Debug Manager] panel are displayed.

- Notes on usage

(1) If this debugging facility is not used, the GTM is handled in the same way as other peripheral IP modules.

(2) If this facility is used, the following facilities are not available.

- LPD output of software tracing
- External trigger input
- External trigger output
- Measuring CAN bus reception processing times
- Pseudo-error Debugging
- Debugging CAN Bus Reception Procedures

(3) The following facilities are not available in debugging of the GTM.

- Software break
- Measure Execution Time of Programs
- Measure Performance
- Set an Action into Programs
- Exclusive Control Checking Tool

(4) The mode selected for the [Debug mode] property in the [Multi-core] category on the [Debug Tool Settings] tabbed page in the Property panel affects some aspects of the operation in the following ways.

- Selecting [Sync debug mode]
  When a channel of the MCS is selected as the target for debugging and [Go to Here] is selected from the context menu, the CPU does not proceed with synchronous execution.
- Selecting [Async debug mode]
  When a CPU is selected as the target for debugging and [Standby] is displayed as the state of the GTM, all channels of the MCS also proceed with synchronous execution. When all CPUs enter the break state while [Standby] is being displayed as the state of the GTM, a synchronous break is also generated for all channels of the MCS.

(5) When a channel of the MCS is selected as the target for debugging, the CPU does not proceed with synchronous stepped execution.

(6) Access-type events cannot be set for the I/O registers.

(7) When the GTM is selected as the target for debugging and [Standby] is displayed as the state of the GTM, the following facilities are not available.

- Displaying or modifying memory
- Displaying or modifying CPU registers
- Setting, editing, and deleting breakpoints
- Executing or stopping programs

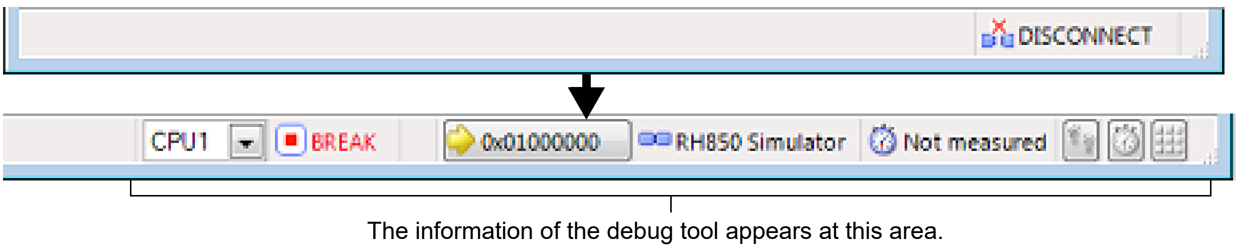
2.5 Connect to/Disconnect from the Debug Tool

This section describes how to connect to/disconnect from the debug tool.

2.5.1 Connect to the debug tool

By selecting [Connect to Debug Tool] from the [Debug] menu, CS+ starts communicating with the debug tool selected in the active project.
After succeeding in the connection to the debug tool, the Statusbar of the Main window changes as follows:
For details on each item displayed on the Statusbar, see the section of the Main window.

Figure 2.9 Statusbar Indicating Successful Connection to Debug Tool



Caution If the version of compiler being used is not supported by CS+, [Connect to Debug Tool] will be disabled.

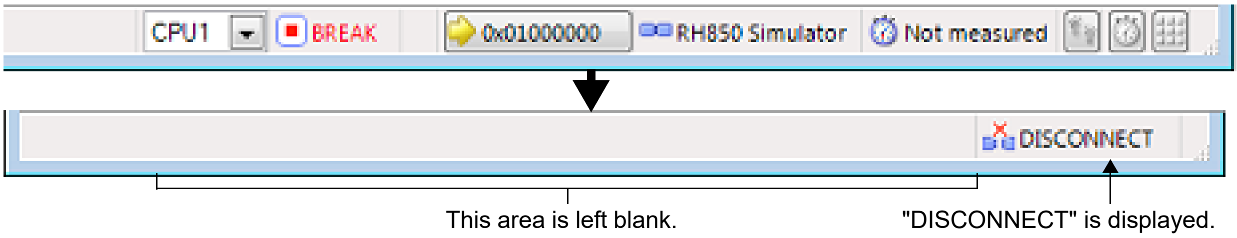
Remark When the button on the Debug toolbar is clicked, the specified file is downloaded automatically after connecting to the debug tool (see "2.6.1 Execute downloading").
When the button on this toolbar is clicked, the project is built automatically, and then the built file is downloaded after connecting to the debug tool.

(1) Display the Emulator information [E1] [E20]
After successful connection to the debug tool, the emulator information such as the LPD clock frequency are displayed on the Output panel.

2.5.2 Disconnect from the debug tool

By clicking the button on the Debug toolbar, CS+ cuts off the communication with the connected debug tool.
After disconnecting from the debug tool, the Statusbar of the Main window changes as follows:

Figure 2.10 Statusbar Indicating Disconnection from Debug Tool



Caution The debug tool cannot be disconnected from CS+ while the program is running.

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

2.5.3 Connect to the debug tool using hot plug-in

With hot plug-in function, you can connect the debug tool to the target board during execution of a program (without having to turn off the system) and debug the program while it is in execution.
Follow the steps below to establish hot plug-in connection.

Caution 1. The hot plug-in connection is enabled only when the on-chip debugging emulator and IE850A is used.

Caution 2. When a hot plug-in connection is made, the settings of the following properties are ignored (i.e. the program operates as if the specification for them is [No]). The settings of them become valid again after reconnection with CS+.

- [Power target from the emulator (MAX 200mA)]
- [Set OPJTAG in LPD connection before connecting]
- [Set OPJTAG in JTAG connection before disconnecting]
- [Using the code flash self programming
- [Change the clock to flash writing]
- [Mask WAIT signal]
- [Mask RESET signal]

Caution 3. When a hot plug-in connection is made, events currently being set in the project are ignored. They become valid again after reconnection with CS+.

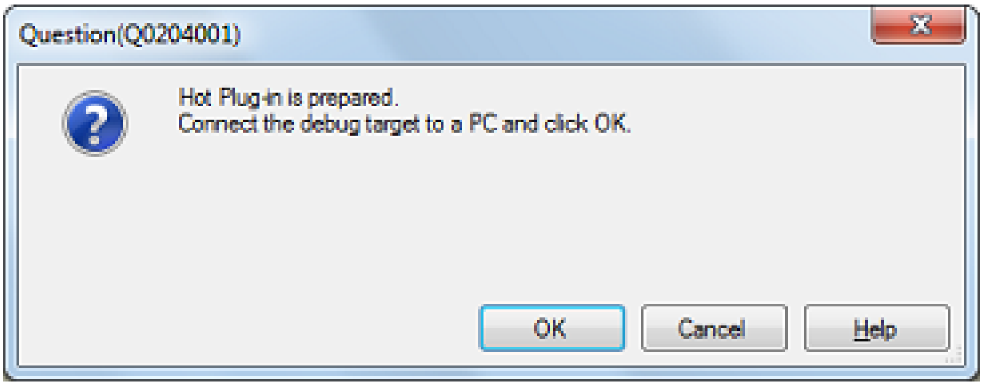
(1) Execute the program
Execute the program which has been downloaded onto the microcontroller on the target board without connecting to the emulator.

(2) Select the debug tool
In the active project, select the debug tool which supports hot plug-in connection (E1/E20).

(3) Connect the debug tool to CS+ using hot plug-in
Select [Hot Plug-in] from the [Debug] menu to initiate the preparation for hot plug-in connection.

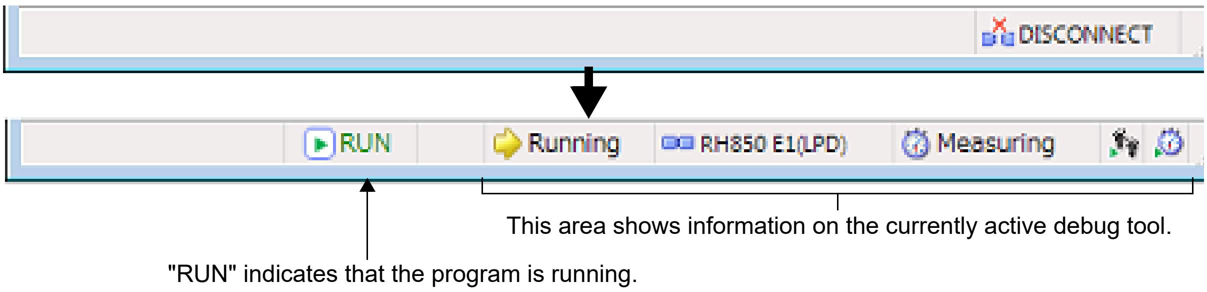
(4) Connect to the target board
Following message will appear once you are ready to start hot plug-in connection. Connect the emulator to the target system and click [OK]. This will start the communication with the debug tool which is selected in the currently active project.

Figure 2.11 Message Indicating that Hot Plug-in Connection Is Ready to Be Started



(5) Hot plug-in connection completed
Once the connection to the debug tool is successfully completed, the Statusbar on the Main window will change as shown below. For details on each item displayed on the statusbar, see the section of the "Main window".

Figure 2.12 Statusbar Indicating Successful Hot plug-in Connection to Debug Tool



2.6 Download/Upload Programs

This section describes how to download programs (such as load module files) to debug to CS+ and how to upload the memory contents being debugged from CS+ to files.

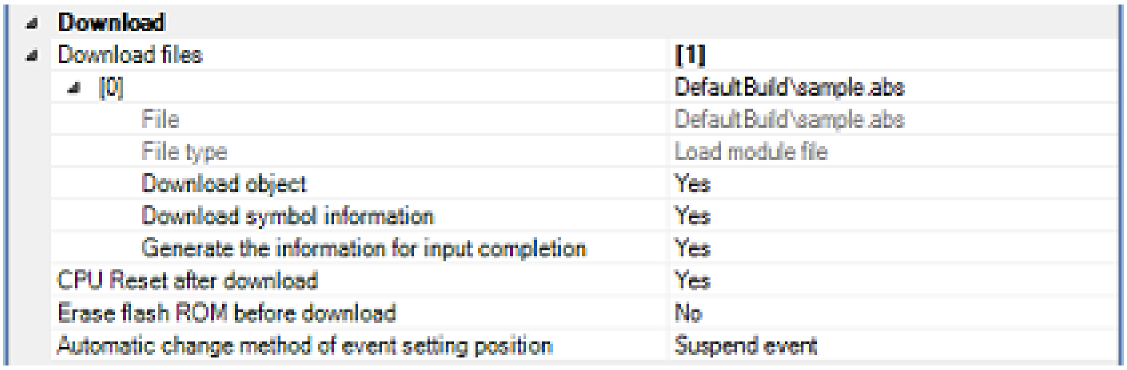
2.6.1 Execute downloading

Download the load module file to be debugged to the debug tool that is currently connected.
Follow the steps below on the [Download File Settings] tab in the Property panel for the downloading, and then execute the downloading.

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.

(1) Setting the [Download] category

Figure 2.13 [Download] Category



(a) [Download files]
The names of files to be downloaded and download conditions are displayed (the number enclosed with "[ ]" indicates the number of files to be download).
Files that are specified as build target files in the main project or subprojects will automatically be selected as the files to be downloaded[Note 1].
However, you can manually change the download files and the condition. In this case, see "2.6.2 Advanced downloading".[Note 2]

Note 1. 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 a download file to Download files node on project tree. The file to be downloaded will be reflected in this property.
See "CS+ Integrated Development Environment User's Manual: Project Operation" for details on the using an external build tool and a debug-dedicated project.

Note 2. The emulator does not support downloading to external flash memory.

(b) [CPU Reset after download]
Specify whether to reset the CPU after downloading.
Select [Yes] to reset the CPU (default).

(c) [Erase flash ROM before download] [Full-spec emulator][E1][E20]
Specify whether to erase the flash ROM before downloading.
Select [Yes] to erase the flash ROM (default: [No]).
If this property is set to [Yes], an erase will be performed in the area where the downloaded data exists.

(d) [Automatic change method of event setting position]
If the file is downloaded again during debugging then the location (address) set for the currently configured event may change to midway in the instruction.
Specify with this property how to handle the target event in this circumstance.
Select one of the options from the following drop-down list.

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).

Note, however, that this property setting only applies to the location setting of events without debugging information. The location setting of events with debug information is always moved to the beginning of the source text line.

(e) [Allow downloading to Configuration Setting Area] [E2][IE850A]
Select whether to allow downloading to the Configuration Setting Area.
Select [Yes] to allow downloading to the Configuration Setting Area (default: [No]).
Reconnect to the debug tools when after downloading to the Configuration Setting Area.
Note that this property is set to [No] after downloading.
This property will not be saved in the project information.

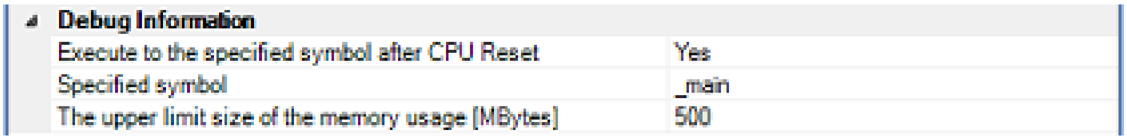
(f) [Allow downloading to Block Protection Area] [E2][IE850A]
Select whether to allow downloading to the Block Protection Area.
Select [Yes] to allow downloading to the Block Protection Area (default: [No]).
Reconnect to the debug tools when after downloading to the Block Protection Area.
Note that this property is set to [No] after downloading.
This property will not be saved in the project information.

(g) [Allow downloading to Security Setting Area] [E2][IE850A]
Select whether to allow downloading to the Security Setting Area.
Select [Yes] to allow downloading to the Security Setting Area (default: [No]).
Reconnect to the debug tools when after downloading to the Security Setting Area.
Note that this property is set to [No] after downloading.
This property will not be saved in the project information.

(h) [Allow downloading to Switch Area] [E2][IE850A]
Select whether to allow downloading to the Switch Area.
Select [Yes] to allow downloading to the Switch Area (default: [No]).
Reconnect to the debug tools when after downloading to the Switch Area.
Note that this property is set to [No] after downloading.
This property will not be saved in the project information.

(2) Setting the [Debug Information]

Figure 2.14 [Debug information] Category



(a) [Execute to the specified symbol after CPU Reset]
Specify whether to execute the program to the specified symbol position after CPU reset or downloading (for only when the [CPU Reset after download] property is set to [Yes]).
Select [Yes] to execute the program to the specified symbol position after CPU reset (default).

Remark When the [CPU Reset after download] property is set to [Yes], the operation after downloading is as follows:

- If [Yes] is selected for this property, the Editor panel will open automatically with displaying source text of the position specified with the [Specified symbol] property after downloading.
- If [No] is selected for this property, the Editor panel will open with displaying source text of the reset address (when if the source text has not been allocated to the reset address, the contents of the reset address is displayed in the Disassemble panel).

(b) [Specified symbol]
This property appears only when the [Execute to the specified symbol after CPU Reset] property is set to [Yes].
Specify the position at which the program is stop after CPU reset.
Directly enter an address expression between 0 and "*last address in address space*" (default: [_main]).
Note, however, that the program will not be executed if the specified address expression cannot be converted into an address.

Remark Normally, specify the following.
For assembly source: Start label corresponding to main function
For C source: Symbol assigned to the start of the main function name

(c) [The upper limit size of the memory usage [Mbytes]]
Specify the upper limit on the amount of memory to be used in reading the debug information.
When the amount of memory being used exceeds the upper limit specified here, memory is made available by discarding debug information that has been read until the amount of memory in use is reduced to half of this upper limit (lowering the upper limit might improve the situation when shortages of memory are arising).
Directly enter a decimal number between 100 and 1000 (unit: Mbyte) (default: [500]).

Caution In some cases, lowering the upper limit may lead to poorer responsiveness since it leads to more frequent discarding and re-reading of debug information.

(3) Executing a download
Click the button on the Debug toolbar.
If this operation is performed while disconnecting from the debug tool, the application automatically connects to the debug tool, and then performs the download.

Remark When a program that has been modified during debugging is re-downloaded, you can easily build and download it by selecting [Build & Download] from the [Debug] menu on the Main window.

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

If the load module file is successfully downloaded, the Editor panel opens automatically, and the contents of the downloaded file's source text are displayed.

Remark You can automatically overwrite the value of I/O register/CPU register with the specified values before and after performing the download (see "2.20 Use Hook Function" for details).

2.6.2 Advanced downloading

You can change the download files and the condition to download.
With CS+, the following file types can be downloaded.

Table 2.4 Downloadable File Formats

Downloadable File	Extension	File Format
Load module file	.abs or no limit Note 1	Load module file format
Intel HEX file	.hex, .run Note 2	Intel HEX file format
Motorola S-record file	.mot, .run Note 2	Motorola S-record file format - (S0, S1, S9-16 bits) - (S0, S2, S8-24 bits) - (S0, S3, S7-32 bits)
Binary file	.bin	Binary file format

Note 1. When the build tool is other than GHS CCRH850, only ".abs" can be specified. When the build tool is GHS CCRH850, there is no restriction on the filename extension.

Note 2. ".run" can be specified only when the build tool is GHS CCRH850.

Caution Notes on using GHS compiler (Green Hills Software, LLC)

- Supported version, supported options, and non-supported Options

Supported Compiler Version	Supported Options				Non-supported Options
	Debug Option	-cpu Option Note 1	Optimization Option	Other	Linker Optimization Option Note 3
Ver.2024.1.4 Ver.2023.5.4	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=rh850g3mh -cpu=rh850g3kh -cpu=rh850g4mh -cpu=rh850g4kh -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt -rh850_abi=ghs2014	-shorten_loads -shorten_moves -delete -codefactor
Ver.2023.1.4 Ver.2022.1.4 Ver.2021.1.5 Ver.2020.5.5 Ver.2020.1.5 Ver.2019.5.5 Ver.2019.1.5 Ver.2018.5.5 Ver.2018.1.5 Ver.2017.5.5 Ver.2017.1.5	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=rh850g3mh -cpu=rh850g3kh -cpu=rh850g4mh -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt -rh850_abi=ghs2014	-shorten_loads -shorten_moves -delete -codefactor
Ver.2016.5.5	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=rh850g3mh -cpu=rh850g3kh -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt -rh850_abi=ghs2014	-shorten_loads -shorten_moves -delete -codefactor
Ver.2015.1.7	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=rh850g3mh -cpu=rh850g3kh -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt	-shorten_loads -shorten_moves -delete -codefactor
Ver.2015.1.5	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt	-shorten_loads -shorten_moves -delete -codefactor
Ver.2014.1.7	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=v850e3	-Odebug -Ogeneral or -O -Osize -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt	-shorten_loads -shorten_moves -delete -codefactor
Ver.2013.5.5	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt	-shorten_loads -shorten_moves -delete -codefactor

Supported Compiler Version	Supported Options				Non-supported Options
	Debug Option	-cpu Option Note 1	Optimization Option	Other	Linker Optimization Option Note 3
Ver.2013.1.5	-G -dual_debug	-cpu=rh850 -cpu=rh850g3k Note 2 -cpu=rh850g3m -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt	-shorten_loads -shorten_moves -delete -codefactor
Ver.2012.5.5	-G -dual_debug	-cpu=rh850 -cpu=v850e3	-Odebug -Ogeneral or -O -Ospeed -Onone -Osize Note 4	-prepare_dispose -callt	-shorten_loads -shorten_moves -delete -codefactor

Note 1. For details, see the release notes of GHS products.

Note 2. -cpu=rh850 is synonymous with -cpu=rh850g3k.

Note 3. Linker changes execution code. However, the changes are not reflected on the debug information.

Note 4. The Disassemble panel may open when the program is stopped after stepped execution or the program may continue to be executed without a halt in stepped execution.

- Notes on debugging
  - The following three methods are available when a load module file from the GHS compiler is in use.
    - Create a debug-dedicated project and add the load module file that has been built.
    - Specify [Empty Application(GHS CCRH850)] as the project type, create, and build a project.
    - Specify [Using Existing GHS Project File(GHS CCRH850)] as the project type, create a project, and add the load module file that has been built.

    For the project type, see "CS+ Integrated Development Environment User's Manual: GHS CCRH850 Build Tool Operation".
  - Select [Green Hills Software] in the [Compiler Vendor] property in the Download Files dialog box to add the load module file.
  - The followings are not supported:
    - C++
    - Programs with C99 own type or modifier
    - Programs with gnu c extensional specs
  - **Step or Execution related functions**
    Executing return out functions from the following function may fail. And call history on the Call Stack panel is not shown incorrect.
    - Functions is called by callt
    - Interrupt Functions
  - Reference function of variables using expressions
    - When long long type or double type variables are located to register, only lower 4 bit register name is shown in address column in the Watch panel. CS+ gets upper 4 bit value from next to lower 4 bit register.
      For example, if R4 is shown in the address column, then CS+ will get the upper 4 bit value from R5.
    - When structure type variables are located to a registers, correct value of structure members aren't shown on the Watch panel. See the value that is show in register of [address] area on the CPU Register panel.
    - Even using an expression with scope specify, it is impossible to refer defined static variables in functions.
      When a program counter exists in the function that the static variables are defined in, it is possible to refer it.

```
func(){
    static  sta = 100;
}
```

In the above case, during debugging func(), it is possible to refer both "sta" and "func()#sta". During debugging functions except func(), it is possible to refer neither "sta" nor "func()#sta".

- As stack frame is not generated at a start point of a function, passed address of a variable via the stack is not correct. Please refer value of a variable after stepping in the function.

- Other than the above
  - It is not possible to invalidate the Symbol name completion function (a specification of [Generate the information for input completion] item in the Download Files dialog box will be ignored).

You can change the download files or download conditions in the following Download Files dialog box.
The Download Files dialog box is opened by clicking the [...] button that appears at the right edge in the column of the [Download files] property when you select it in the [Download] category on the [Download File Settings] tab of the Property panel.

Figure 2.15 Opening Download Files Dialog Box

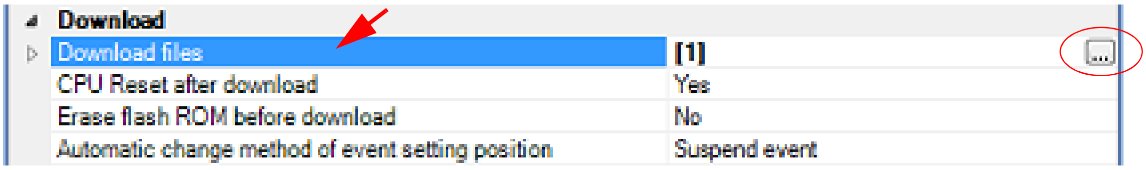
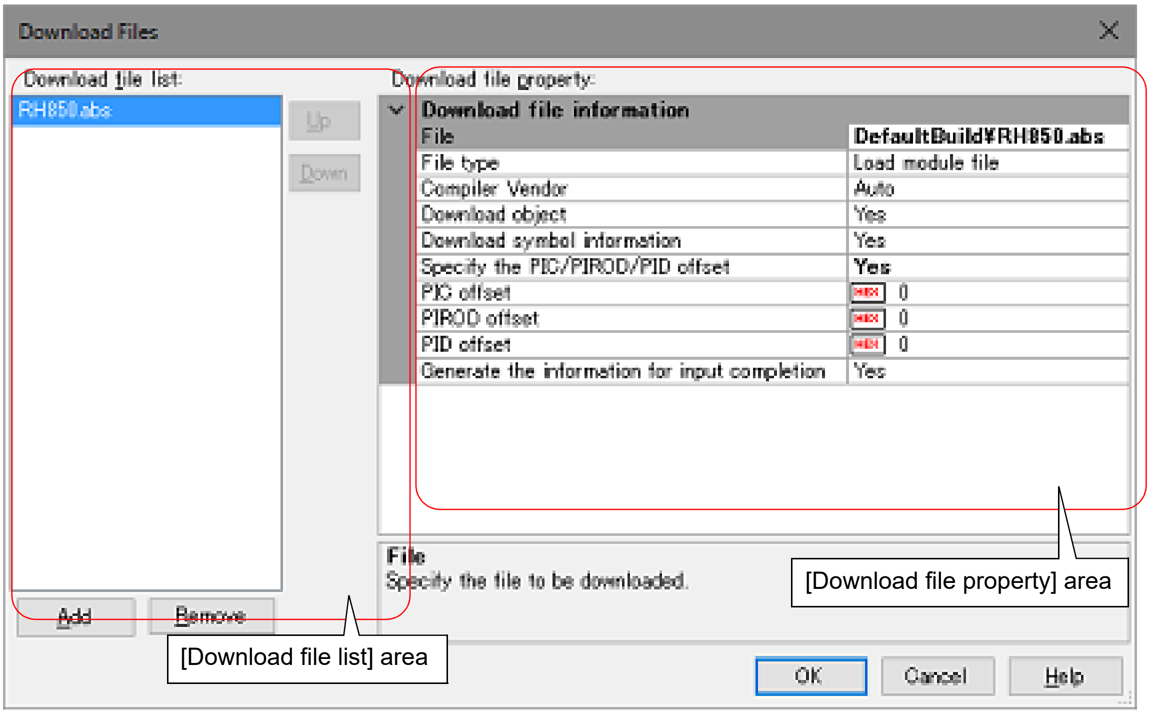


Figure 2.16 Advanced Downloading (Download Files Dialog Box)



This section describes how to configure on the Download Files dialog box above when the following cases.

2.6.2.1 Change download conditions for load module files
2.6.2.2 Add download files (*.hex/*.mot/*.bin)
2.6.2.3 Download multiple load module files
2.6.2.4 Perform source level debugging with files other than the load module file format

2.6.2.1 Change download conditions for load module files

Follow the steps below in the Download Files dialog box to change the download conditions (object information and symbol information) for load module files.

(1) Select a load module file
Select a load module file to download in the [Download file list] area.

(2) Change download conditions
Current download conditions for the selected load module file are displayed in the [Download file property] area. Change each items displayed in the property.

Download object	Select whether to download the object information from the specified file.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloads object information.
		No	Does not download object information.
Download symbol information	Select whether to download the symbol information from the specified file[Note 1].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloads symbol information.
		No	Does not download symbol information.
Specify the PIC/ PIROD/PID offset	Specify whether to change the positions of PIC (Position Independent Code), PIRDD (Position Independent Read Only Data) 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", "PIROD Offset" and "PID Offset" will appear as sub-items.		
	Default	No	
	How to change	Select from the drop-down list.	
	Specifiable value	Yes	PIC/PIROD/PID offset is specified[Note 2].
		No	PIC/PIROD/PID offset is not specified.
PIC Offset	Input the offset values from the start address of the program section specified at the time of load module creation. For example, if 1000 is specified for this item when a load module for which the program section starts at address 1000 is to be downloaded, the section will be downloaded to address 2000.		
	Default	0	
	How to change	Enter directly from the keyboard.	
	Specifiable value	Hex number between 0x0 and 0xFFFFFFFF	

PIROD Offset	Input the offset values from the start address of the ROM data section specified at the time of load module creation. For example, if 1000 is specified for this item when a load module for which the ROM data section starts at address 1000 is to be downloaded, the section will be downloaded to address 2000.		
	Default	0	
	How to change	Enter directly from the keyboard.	
	Specifiable value	Hex number between 0x0 and 0xFFFFFFFF	
PID Offset	Input the offset values from the start address of the RAM data section specified at the time of load module creation. For example, if 1000 is specified for this item when a load module for which the RAM data section starts at address 1000 is to be downloaded, the section will be downloaded to address 2000.		
	Default	0	
	How to change	Enter directly from the keyboard.	
	Specifiable value	Hex number between 0x0 and 0xFFFFFFFF	
Generate the information for input completion	Select whether to generate the information for the Symbol name completion function when downloading[Note 3].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Generates the information for the symbol name completion function. (i.e. uses the symbol name completion function.)
		No	Does not generate the information for the symbol name completion function. (i.e. does not use the symbol name completion function.)

Note 1. If the symbol information have not been downloaded, the source level debugging cannot be performed.

Note 2. Proper debug operation is not guaranteed when you have selected "Yes" for load modules that were created without using PIC/PID function (see Section "2.8 Usage of PIC/PID Function").

Note 3. When [Yes] is selected, the time taken for downloading and the memory usage on the host machine will increase. We recommend selecting [No] in this item if you do not intend to use the symbol name completion function.

(3) Click the [OK] button
Enable all the configuration in this dialog box and change the download conditions.

2.6.2.2 Add download files (*.hex/*.mot/*.bin)

Follow the steps below to add download files other than the load module file format (Intel HEX file (*.hex), Motorola S-record file (*.mot), or binary file (*.bin)) in the Download Files dialog box.

(1) Click the [Add] button
When the [Add] button is clicked, a blank list item "-" is displayed in the last line of the [Download file list] area.

(2) Property configuration of the download files to add
Configure the download conditions for the download file to add in the [Download file property] area.
Configure each item displayed with the following condition. When the configuration is completed, the file name specified in this property is displayed in the blank list of the [Download file list] area.

<table>
<tr><td rowspan="4">File</td><td colspan="3">Specify the download file (Intel HEX file (*.hex), Motorola S-record file (*.mot), or binary file (*.bin)) to download (up to 259 characters).</td></tr>
<tr><td>Default</td><td colspan="2">Blank</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard, or specify with the Select Download File dialog box opened by clicking the [...] button.</td></tr>
<tr><td>Available values</td><td colspan="2">See "Table 2.4 Downloadable File Formats".</td></tr>
<tr><td rowspan="7">File type</td><td colspan="3">Select the type of the file to download.
Here, select a item other than [Load module file].</td></tr>
<tr><td>Default</td><td colspan="2">Load module file</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="4">Available values</td><td>Load module file</td><td>Specifies a load module file.</td></tr>
<tr><td>Hex file</td><td>Specifies an Intel HEX file.</td></tr>
<tr><td>S record file</td><td>Specifies a Motorola S-record file.</td></tr>
<tr><td>Binary data file</td><td>Specifies an binary file.</td></tr>
<tr><td rowspan="4">Offset</td><td colspan="3">Specify the offset from the address at which the file's download is to start.
Note that this item appears only when [File type] is set to [Hex file] or [S record file].</td></tr>
<tr><td>Default</td><td colspan="2">0</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">0x0 to 0xFFFFFFFFFF in hexadecimal number</td></tr>
<tr><td rowspan="4">Start address</td><td colspan="3">Specify the address at which to start the file's download.
Note that this item appears only when [File type] is set to [Binary file].</td></tr>
<tr><td>Default</td><td colspan="2">0</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">0x0 to 0xFFFFFFFFFF in hexadecimal number</td></tr>
</table>

Remark The settings of whether to download the object information or symbol information can be made only when the type of the file to download is load module files.

(3) Check the order of download
The order of the download is the display order of the files displayed in the [Download file list] area.
If you want to change the order, use the [Up]/[Down] button.

(4) Click the [OK] button
Enable all the configuration in this dialog box and add a download file (the file name is displayed in the [Download] category on the [Download File Settings] tab of the Property panel).

2.6.2.3 Download multiple load module files

Follow the steps below on the Download Files dialog box to download multiple load module files.

Caution When debugging a program consisting of multiple load module files, care should be taken to avoid overlapping of location addresses.

(1) Click the [Add] button
When the [Add] button is clicked, a blank list item "-" is displayed in the last line of the [Download file list] area.

(2) Property configuration of the download files to add
Configure the download conditions for the download file to add in the [Download file property] area.

Configure each item displayed with the following condition.
When the configuration is completed, the file name specified in this property is displayed in the blank list of the [Download file list] area.

File	Specify the name of the load module file to be added (up to 259 characters).		
	Default	Blank	
	Modifying	Directly enter from the keyboard, or specify with the Select Download File dialog box opened by clicking the [...] button displayed at the right edge of this property when it is selected.	
	Available values	See "Table 2.4 Downloadable File Formats".	
File type	Specify the type of the file to download. Here, select [Load module file].		
	Default	Load module file	
Download object	Select whether to download the object information from the specified file.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloads object information.
		No	Does not download object information.
Download symbol information	Select whether to download the symbol information from the specified file[Note 1].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloads symbol information.
		No	Does not download symbol information.
Generate the infor-mation for input completion	Select whether to generate the information for the Symbol name completion function when downloading[Note 2].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Generates the information for the symbol name completion function. (i.e. uses the symbol name completion function.)
		No	Does not generate the information for the symbol name completion function. (i.e. does not use the symbol name completion function.)

Note 1. If the symbol information have not been downloaded, the source level debugging cannot be performed.

Note 2. When [Yes] is selected, the time taken for downloading and the memory usage on the host machine will increase. We recommend selecting [No] in this item if you do not intend to use the symbol name completion function.

Remark You can decrease the memory usage by selecting [No] for the [Download symbol information] item if the symbol information is not required for the module (in this case, however, the source level debugging can not be performed for the file).

(3) Check the order of download
The order of the download is the display order of the files displayed in the [Download file list] area.
If you want to change the order, use the [Up]/[Down] button.

(4) Click the [OK] button

Enable all the configuration in this dialog and add the specified load module file (the specified file name is displayed in the [Download] category on the [Download File Settings] tab of the Property panel).

2.6.2.4 Perform source level debugging with files other than the load module file format

Even when an Intel HEX file (*.hex), Motorola S-record file (*.mot), or binary file (*.bin) is specified to be the subject file to download, it is possible to do source level debugging by downloading symbol information for the load module file from which the subject file was created, along with the subject file that you download.

To do so, follow the steps below on the Download Files dialog box.

(1) Click the [Add] button
When the [Add] button is clicked, a blank list item "-" is displayed in the last line of the [Download file list] area.

(2) Property configuration of the load module file to add
Configure each item displayed with the following condition in the [Download file property] area.

<table>
<tr><td>File</td><td colspan="3">Specify a load module file from which the Intel HEX file (*.hex), Motorola S-record file (*.mot), or binary file (*.bin) that you want to download was created.
Directly enter from the keyboard, or specify with the Select Download File dialog box opened by clicking the [...] button that appears at right by selecting this property.</td></tr>
<tr><td>File type</td><td colspan="3">Select [Load module file] (default).</td></tr>
<tr><td>Download object</td><td colspan="3">Select [No].</td></tr>
<tr><td>Download symbol information</td><td colspan="3">Select [Yes] (default).</td></tr>
<tr><td rowspan="5">Generate the information for input completion</td><td colspan="3">Select whether to generate the information for the Symbol name completion function when downloading[Note].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Generates the information for the symbol name completion function. (i.e. uses the symbol name completion function.)</td></tr>
<tr><td>No</td><td>Does not generate the information for the symbol name completion function. (i.e. does not use the symbol name completion function.)</td></tr>
</table>

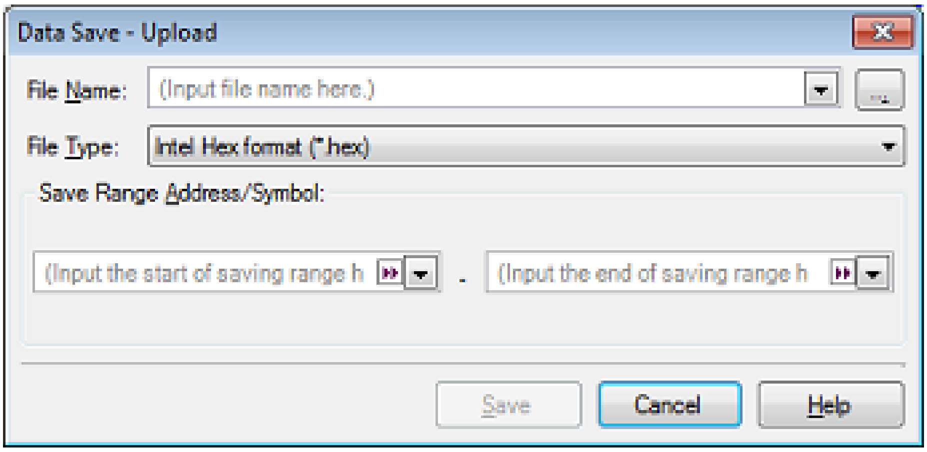
Note When [Yes] is selected, the time taken for downloading and the memory usage on the host machine will increase. We recommend selecting [No] in this item if you do not intend to use the symbol name completion function.

(3) Click the [OK] button
Enable all the configuration in this dialog box and add the specified load module file (Only the symbol information included in the load module file will be downloaded).

2.6.3 Execute uploading

The contents of the memory of the debug tool currently connected can be saved (uploaded) in an arbitrary file.
You can upload the data in the Data Save dialog box that is opened by selecting the [Debug] menu >> [Upload...].
In this dialog box, follow the steps below.

Figure 2.17 Execute Uploading (Data Save Dialog Box)



(1) Specify [File Name]
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 [...] button, and selecting a file via the Select Data Save File dialog box.

(2) Specify [File Type]
Select the format in which to save the file from the following drop-down list.
The following file formats can be selected.

Table 2.5 Uploadable File Formats

List Display	File Format
Intel Hex format (*.hex)	Intel HEX file format (The expanded linear address record is always used)
Motorola S-record (*.mot)	Motorola S-record file format
Binary data (*.bin)	Binary file format

(3) Specify [Save Range Address/Symbol]
Specify the range of addresses to save via "start address" and "end addresses".
Directly enter hexadecimal number/address expression in each text box or select from the input history displayed in the drop-down list (up to 10 items).

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

(4) Click the [Save] button
Save the contents of the memory in the specified file in specified format as upload data.

2.7 Display/Change 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.

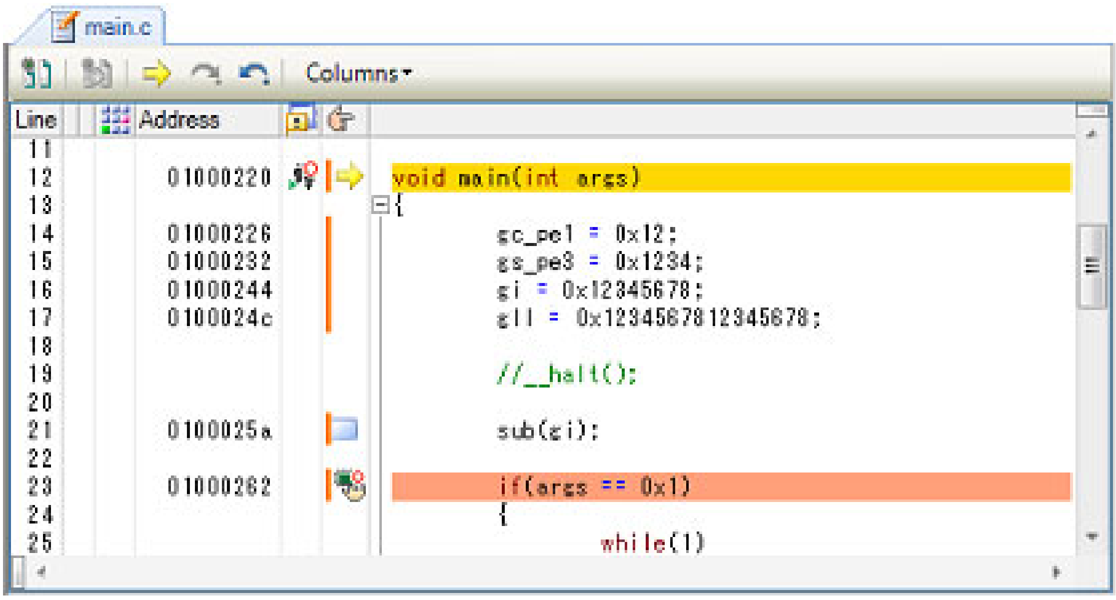
- Editor panel
  The source file is displayed and can be edited.
  Furthermore, the source level debugging/instruction level debugging (see "2.10.3 Execute programs in steps") and the display of the code coverage measurement result **[Simulator]** (see "2.17 Measure Coverage [Simulator]") can be performed 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 "2.10.3 Execute programs in steps") and the display of the code coverage measurement result **[Simulator]** (see "2.17 Measure Coverage [Simulator]") can be performed in this panel. In this panel, the disassemble results can be displayed with the corresponding source text (default).

Remark It is normally necessary to download a load module file with debugging information in order to perform the source level debugging, but it is also possible to do so by downloading an Intel HEX file (*.hex), Motorola S-record file (*.mot), or binary file (*.bin) (see "2.6.2.4 Perform source level debugging with files other than the load module file format").

2.7.1 Display source files

The source file is displayed in the Editor panel below. The Editor panel automatically opens with displaying source text of the specified position (see "2.6.1 Execute downloading") when a load module file is successfully downloaded.
If you want to open the Editor panel manually, double-click on the source file in the Project Tree panel.
For details on the contents and function in each area, see the section for the Editor panel.

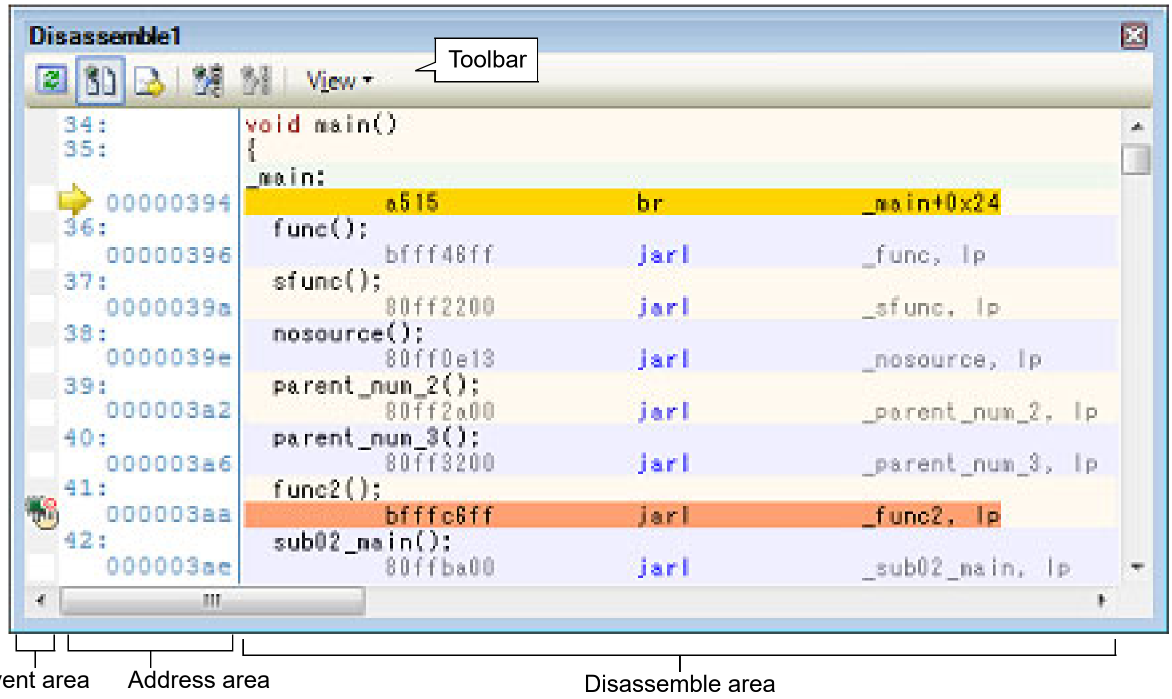
Figure 2.18 Display Source File (Editor Panel)



2.7.2 Display the result of disassembling

The result of disassembling the downloaded program (disassembled text) is displayed in the Disassemble panel below.
Select [View] menu >> [Disassemble] >> [Disassemble*1 - 4*].
The maximum of 4 Disassemble panels can be opened. Each panel is identified by the names "Disassemble1", "Disassemble2", "Disassemble3" and "Disassemble4" on the titlebar.
For details on the contents and function in each area, see the section for the Disassemble panel.

Figure 2.19 Display Result of Disassembling (Disassemble Panel)



Remark You can set the scroll range of the vertical scroll bar on this panel via the Scroll Range Settings dialog box which is opened by clicking the button from [View] on the toolbar.

This section describes the following.

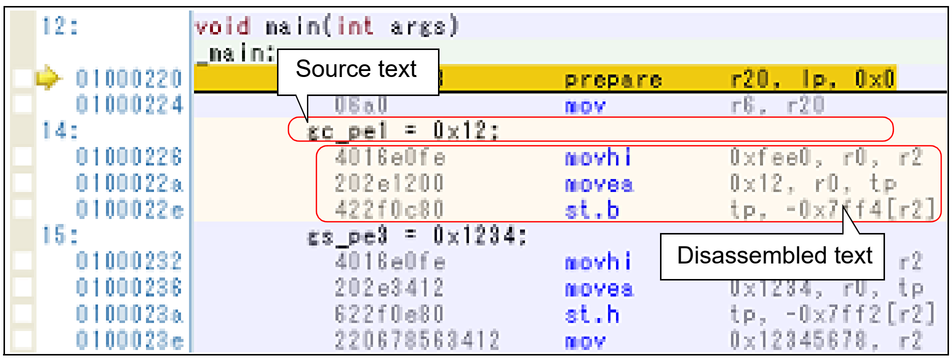
- 2.7.2.1 Change display mode
- 2.7.2.2 Change display format
- 2.7.2.3 Move to the specified address
- 2.7.2.4 Move to the symbol defined location
- 2.7.2.5 Save the disassembled text contents

2.7.2.1 Change display mode

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

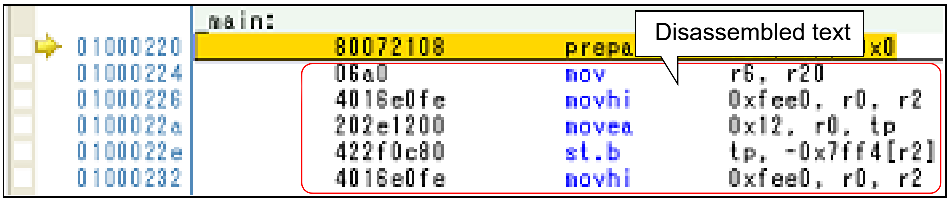
- Mixed display mode
  In this display mode (default), the disassembled text is displayed combined with the source text.

Figure 2.20 Mixed Display Mode (Disassemble Panel)



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

Figure 2.21 Disassemble Display Mode (Disassemble Panel)



2.7.2.2 Change display format

The display format of the disassemble area can be changed using buttons below on the toolbar.

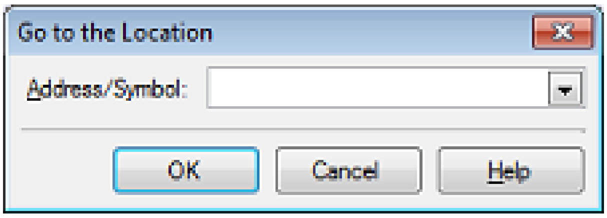
View	The following buttons to change the display format are displayed.
Show Offset	Displays the offset value of the label. The offset value from the nearest label is displayed when a label is defined for the address.
Show Symbol	Displays the address value as the result of disassembling in the format "symbol + offset value" (default). Note that when a symbol has been defined as the address value, only the symbol is displayed.
Show Function Name	Displays the name of the register by its function name (default).
Show Absolute Name	Displays the name of the register by its absolute name.

2.7.2.3 Move to the specified address

You can move to the specified address in the disassembled text in the Go to the Location dialog box which opens when selecting [Go to...] from the context menu.

In this dialog box, follow the steps below.

Figure 2.22 Move to Specified Address in Disassembled Text (Go to the Location Dialog Box)



(1) Specify [Address/Symbol]
Specify the address you want to move the caret to.
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).

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (see "2.21.2 Symbol name completion function").

(2) Click the [OK] button
Caret is moved to the specified address.

2.7.2.4 Move to the symbol defined location

You can move the caret to the address where the symbol is defined.

Click the button on the toolbar after moving the caret to the instruction which refers to the symbol.

Furthermore, click the button on the toolbar following the previous operation returns the caret to the instruction which refers to the symbol at previous caret is defined.

2.7.2.5 Save the disassembled text contents

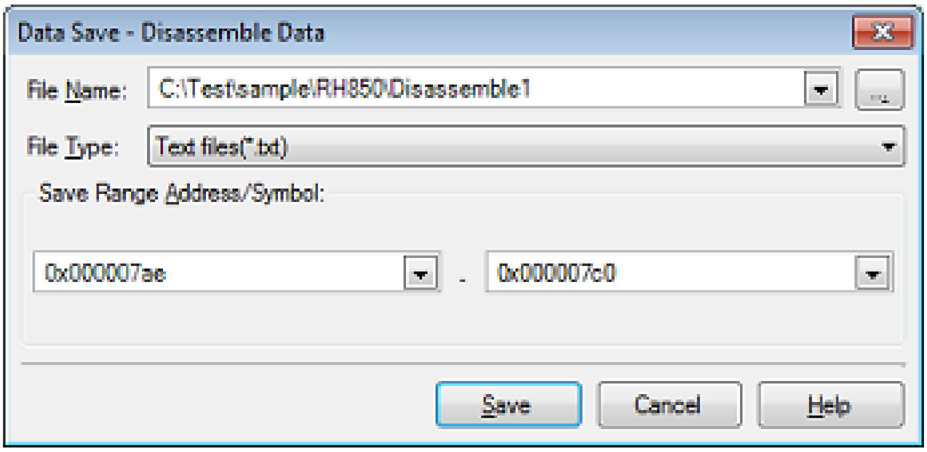
Contents of the disassembled text can be saved in text files (*.txt)/CSV files (*.csv).

When saving to the file, the latest information is acquired from the debug tool, and it is saved in accordance with the display format on this panel.

The Data Save dialog box can be opened by selecting the [File] menu >> [Save Disassemble Data As...] (when this operation takes place with the range selected on the panel, the disassembled data can be saved only for the selected range).

In this dialog box, follow the steps below.

Figure 2.23 Save Disassembled Text Contents (Data Save Dialog Box)



(1) Specify [File Name]
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 [...] button, and selecting a file via the Select Data Save File dialog box.

(2) Specify [File Type]
Select the format in which to save the file from the following drop-down list.
The following file formats can be selected.

List Item	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) Specify [Save Range Address/Symbol]
Specify the range of addresses to save via "start address" and "end addresses".
Directly enter hexadecimal number/address expression in each text box or select from the input history displayed in the drop-down list (up to 10 items).
If a range is selected in the panel, that range is specified as the default. If there is no selection, then the range currently visible in the panel is specified.

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

(4) Click the [Save] button
Disassembling data is saved in the specified file with the specified format.

Figure 2.24 Output Example of Disassembling Data

```
--------------------------------------------------------
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. When the contents of the panel are overwritten and saved by selecting the [File] menu >>[Save Disassemble Data], the Disassemble panels (Disassemble1-4) are handled individually for these respectively. In addition, saving range is same as the previously specified address range.

Remark 2. You can print the current screen image of this panel by selecting the [File] menu >> [Print...].

2.7.3 Run a build in parallel with other operations

CS+ can automatically start a build when one of the following events occurs (rapid build function).

- For other than the debug-only project
  - When any one of the following files that are added to the project is updated:
    (C source file, assembly source file, header file, link directive file, symbol information file, object module file, or library file)
  - When a build target file has been added to or removed from the project
  - When the link order of object module files and library files is changed
  - When the property of the build tool or the build target file is changed
- For the debug-only project
  - When you have edited and saved the C source file, assembly source file and header file that are added to the debug-dedicated project
  - When a C source file, assembly source file, or header file has been added to or removed from the debug-dedicated project
  - When the property of the debug-dedicated project is changed

If a rapid build is enabled, it is possible to perform a build in parallel with the above operations.
To enable/disable a rapid build, select [Rapid Build] from the [Build] menu. A rapid build is enabled by default.

Caution When an external text editor is used, check the [Observe registered files changing] check box on the [General - Build/Debug] category in the Option dialog box to enable this function.

Remark 1. After editing source files, it is recommend to save frequently by pressing the [Ctrl] + [S] key.

Remark 2. Enable/Disable setting of the rapid build applies to the entire project (main project and subprojects).

Remark 3. If you disable a rapid build while it is running, it will be stopped at that time.

2.7.4 Perform line assembly

Instructions and code displayed in the Disassemble panel can be edited (line assembly).
This section describes the following.

2.7.4.1 Edit instructions

Follow the steps below to edit instructions.

(1) Switch to edit mode
Double-click the instruction to edit or select [Edit Disassemble] from the context menu after moving the caret to the instruction to edit.

(2) Edit instructions
Use keyboard to directly edit the instructions.

(3) Write to memory
Press the [Enter] key to line assemble the edited instructions after editing. The code is automatically written to the memory.
If the edited instruction is invalid, the instruction is shown in red and will not be written to the memory.

If there is a space because of overwriting the displayed result of disassembling by another instruction, its byte number is automatically compensated with nop instruction as follows:

Example 1. Overwriting the prepare instruction (8-byte instruction) with the jarl instruction (4-byte instruction)

Before editing	`0432 mov 0x4, r6`
	`1d38 mov r29, r7`
	`8f071b0effff0000 prepare r20, r21, r22, 0x1c, 0x0000ffff`
	`0132 mov 0x1, r6`

After editing	`0432 mov 0x4, r6`
	`1d38 mov r29, r7`
	`bfffe265 jarl 0x100, lp`
	`0000 nop`
	`0000 nop`
	`0132 mov 0x1, r6`

Example 2. Overwriting the mov instruction (2-byte instruction) with the jarl instruction (4-byte instruction)

Before editing	`0432 mov 0x4, r6`
	`1d38 mov r29, r7`
	`8f071b0effff0000 prepare r20, r21, r22, 0x1c, 0x0000ffff`
	`0132 mov 0x1, r6`

After editing	`0432 mov 0x4, r6`
	`bfffe265 jarl 0x100, lp`
	`0000 nop`
	`0000 nop`
	`0000 nop`
	`0132 mov 0x1, r6`

Caution Handling the prepare instruction and dispose instruction
The following table shows the instruction formats of the prepare instruction and dispose instruction. The operand "list12" comprises 12-bit value where a different register is assigned to each bit.

Instruction format of the prepare instruction	`prepare list12, imm5`
	`prepare list12, imm5, sp/imm`
Instruction format of the dispose instruction	`dispose imm5, list12`
	`dispose imm5, list12, [reg1]`

When displaying the results of disassembling the prepare instruction and dispose instruction in the Disassemble panel, the corresponding register names for the operand "list12" are displayed instead of its values as shown in the following examples.

Example 1. When the code is "0xbf, 0x07, 0xe1, 0xff" (4-byte prepare instruction)

View	`prepare r20, r21, r22, r23, r24, r25, r26, r27,r28, r29, r30, r31, 0x20`

Syntax	`prepare   0xfff, 0x20`

Example 2. When the code is "0x90, 0x07, 0xbb, 0xaa 0xff, 0xff, 0xff, 0xff" (8-byte prepare instruction)

View	`prepare   r20, r22, r24, r26, r28, r31, 0x20, 0x7fffffff`
Syntax	`prepare   0x555, 0x20, 0x7fffffff`

Example 3. When the code is "0x51, 0x06, 0xe0, 0xff" (4-byte dispose instruction)

View	`dispose   0x20, r20, r21, r22, r23, r24, r25, r26, r27, r28, r29, r30, r31`
Syntax	`dispose   0x20, 0xfff`

Example 4. When the code is "0x50, 0x06, 0xaa, 0xaa" (4-byte dispose instruction)

View	`dispose   0x20, r20, r22, r24, r26, r28, r31, [r10]`
Syntax	`dispose   0x20, 0x555, [r10]`

Note, however, that it is possible to specify both the value and the register name for the operand "list12" when line assembling the prepare instruction and dispose instruction.

Example 5. In both of the cases (1) and (2) below, the same set of values "0x91, 0x07, 0xe1, 0xff" will be generated as a result of line assembly.

(1)	`prepare   r20, r21, r22, r23, r24, r25, r26, r27,r28, r29, r30, r31, 0x20`
(2)	`prepare   0xfff, 0x20`

Example 6. In both of the cases (1) and (2) below, the same set of values "0xbe, 0x07, 0xbb, 0xaa 0xff, 0xff, 0xff, 0x7f" will be generated as a result of line assembly.

(1)	`prepare   r20, r22, r24, r26, r28, r31, 0x20, 0x7fffffff`
(2)	`prepare   0x555, 0x20, 0x7fffffff`

Example 7. In both of the cases (1) and (2) below, the same set of values "0x51, 0x06, 0xe0, 0xff" will be generated as a result of line assembly.

(1)	`dispose   0x20, r20, r21, r22, r23, r24, r25, r26, r27, r28, r29, r30, r31`
(2)	`dispose   0x20, 0xfff`

Example 8. In both of the cases (1) and (2) below, the same set of values "0x50, 0x06, 0xaa, 0xaa" will be generated as a result of line assembly.

(1)	`dispose   0x20, r20, r22, r24, r26, r28, [r10]`
(2)	`dispose   0x20, 0x555, [r10]`

2.7.4.2 Edit code

Follow the steps below to edit code.

(1) Switch to edit mode
Double-click the code to edit or select [Edit Code] from the context menu after moving the caret to the code to edit.

(2) Edit code
Use keyboard to directly edit the code.

(3) Write to memory
Press the [Enter] key to write the code to the memory after editing.
If the edited instruction is invalid, the instruction is shown in red and will not be written to the memory.
When the code is written to the memory, the result of disassembling is also updated.

2.8 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.
This section describes debugging of a program (load module) whose code or data 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 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.
- PIROD
When the pirod option is specified for compilation, the PIROD function is enabled and the data in constant data areas becomes PIROD. The PIROD always uses PC relative mode to acquire constant data accesses or 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 data areas becomes PID. The PID always uses GP or EP relative mode to acquire data accesses or addresses, so it can be reallocated to any desired addresses even after linkage.

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

Remark 2. For setting of the PIC/PID function by the build tool, see the description of the corresponding properties in "A. WINDOW REFERENCE" in "CS+ Integrated Development Environment User's Manual: CC-RH Build Tool Operation".

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

(1) Set conditions for downloading of the load module file
Specify the offset values ([PIC Offset], [PIROD Offset], or [PID Offset]) from the address of the load module when the load module is created in the code, constant data, or data areas.

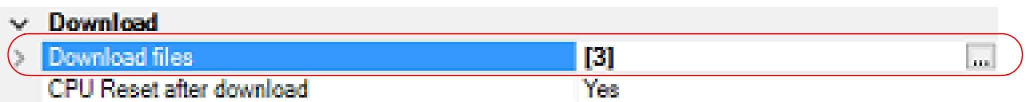
(2) Set a value for the load module file
When the address or offset value is read from memory while the load module is being executed from the reset vector or startup routine, set the value to be read in the target memory.

(3) Download
Download the load-module file (see "2.6.1 Execute downloading").
Debugging of the code and data allocated to new addresses is now possible.

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

The allocation of a load module can be changed through the [Download files] property under the [Download] category on the [Download File Settings] tab in the Property panel.

Figure 2.25 [Download] Category

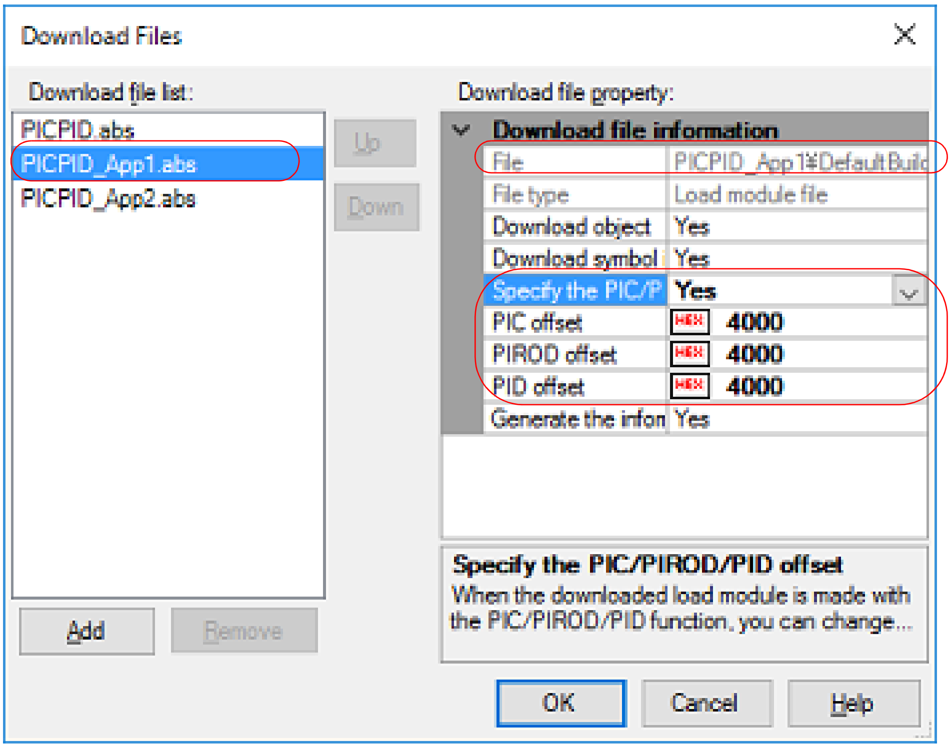


- [Download files]
Click on the [...] button to open the Download Files dialog box.

Set up information on the load module in the [Download file property] area in the [Download file list] area of the Download Files dialog box.

- [File]
  Specify the load-module file using the PIC/PID function.
- [Specify the PIC/PIROD/PID offset]
  Select [Yes]. [PIC Offset], [PIROD Offset] and [PID Offset] will appear.
- [PIC Offset]
  Specify an offset in the code section from the original address allocated at the time the load module was created.
- [PIROD Offset]
  Specify an offset in the constant section from the original address allocated at the time the load module was created.
- [PID Offset]
  Specify an offset in the data section from the original address allocated at the time the load module was created.

Figure 2.26 Adding a Download File and Changing Conditions for Downloading (Download Files Dialog Box)

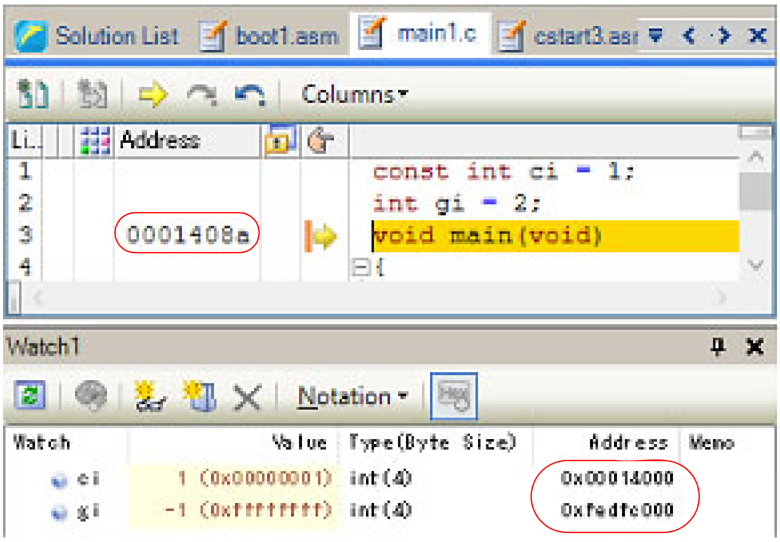


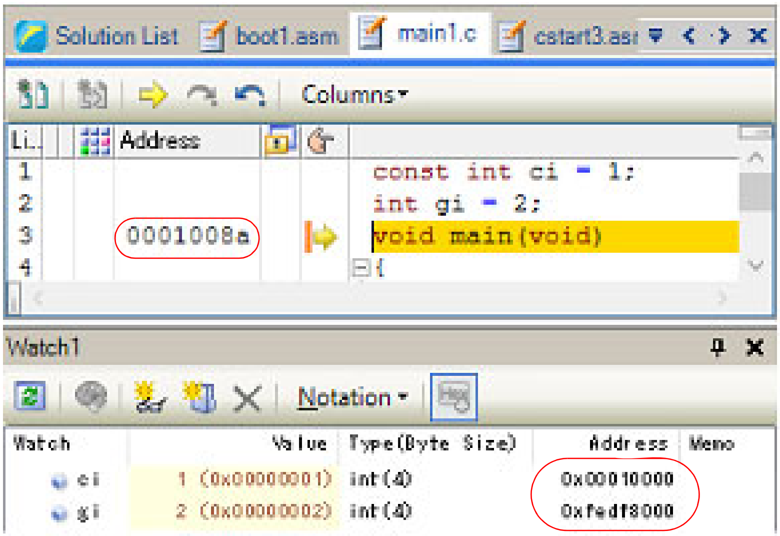
When the load module is downloaded after the values of [PIC Offset], [PIROD Offset] and [PID Offset] have been changed, the allocation of addresses for the code section and external or static variables is changed as shown in the figure below.

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 2.27 Example of Downloading after the Offset Values of [PIC Offset], [PIROD Offset] and [PID Offset] Have been Changed





2.9 Select a Core (PE)

This section describes how to select the target core (PE: Processer Element) to be debugged when the selected microcontroller supports multi-core.

CS+ displays information regarding a core (PE) by switching selection between the target cores to be debugged (see "2.9.1 Switching between cores (PEs)"). Multiple panels are not provided to display each PE in a single dedicated panel.

The following shows the behavior of each CS+ function for a microcontroller that supports multi-core.

Note Selecting an item other than a core (PE) as the target for debugging may cause some facilities to operate in a different way.

(1) Program execution control

(a) [Full-spec emulator][E1][E20]
The control method differs depending on the mode selected by the [Debug mode] property in the [Multi-core] category on the [Debug Tool Settings] tab of the Property panel.

- When [Sync debug mode] is selected:
  Synchronous execution and synchronous break are available in all PEs in principle.
  In step execution, one instruction each is executed in units of instructions.

- When [Async debug mode] is selected:
  Only the PE selected to be debugged is executed and a break is generated.

Caution Step execution is available only in the currently selected PE.
Step execution in source level units, however, may also proceed on other PEs (PE).

(b) [Simulator]
Synchronous execution and synchronous break are available in all PEs in principle.
Step execution is performed in synchronization with the operating frequency.

(2) Event occurrence
All events excluding software break events are set in only the PE selected to be debugged.
However, when setting a software break event in an internal RAM area for which "(Self)" is added to the memory type, it can be set in only an area of the PE that was selected to be debugged.
The Events panel displays a list of events set in all PEs.
Other panels display only events set in the PE selected to be debugged.

(3) Information of the memory, registers, or variables

(a) Memory mappings
Memory mappings may differ depending on the currently selected PE.
In this case, switching to another PE displays the corresponding memory mappings in the [Memory] category on the [Debug Tool Settings] tab of the Property panel or the Memory Mapping dialog box.

(b) Range and values of memory
The same value is displayed or set regardless of which PE is currently selected.
In the Local RAM self area, note that the value in the currently selected PE is acquired and displayed or set.

(c) Register values (including IOR/PC)
The value in the currently selected PE is acquired and displayed or set.

(d) Symbols (including watch-expressions/variable names)
The address and value are determined based on the PC value in the currently selected PE (for example, even when a symbol is valid only in a certain PE, its address and value are determined based on the PC value in the currently selected PE).

(e) Call stack information
The value in the currently selected PE is acquired and displayed or set.

(4) Other functions

(a) Collection of execution history of programs

- **[Full-spec emulator][E1][E20]**
The operation differs depending on the specification of the [Trace target] property in the [trace] category on the [Debug Tool Settings] tab of the Property panel.

  - When [Debug core only] is selected:
    Trace data regarding the currently selected PE is collected.
    Therefore, to collect desired trace data, select the target PE before executing the program (if PE is switched after collecting trace data, the display in the Trace panel will not be updated).

- When [All core] is selected:
  Trace data is collected in all PEs.
  After collecting trace data, switching to another PE displays the corresponding trace data in the Trace panel.
- When *core name* is selected:
  Trace data for the selected *core name* is collected.

- **[Simulator]**
  Trace data regarding the currently selected PE is collected.
  Therefore, to collect desired trace data, select the target PE before executing the program (if PE is switched after collecting trace data, the display in the Trace panel will not be updated).

(b) Measurement of execution time of programs
The execution time is measured in all PEs.
After measurement, switching to another PE displays the corresponding measurement time.

(c) Coverage measurements
The coverage is measured for access in all PEs.
In the Local RAM self area, note that the measurement results will be displayed only for the access in the currently selected PE.

(d) Performance measurements
The performance is measured in all PEs.
After measurement, switching to another PE displays the corresponding measurement result.

2.9.1 Switching between cores (PEs)

The core (PE) to be debugged can be selected in either of the following two ways.

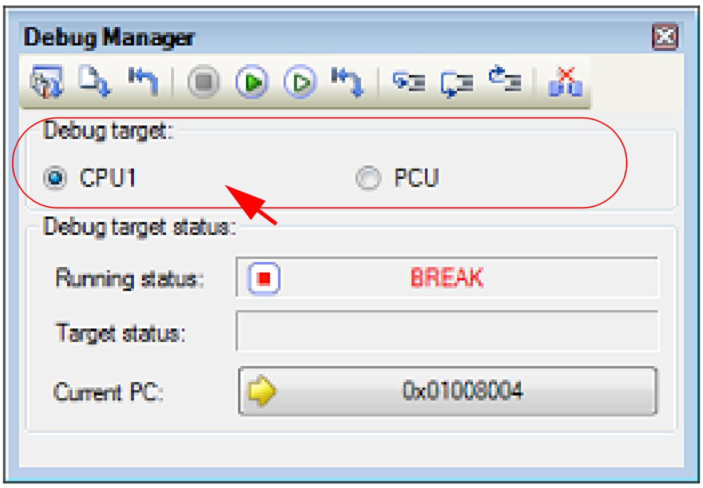
(1) Switching through the statusbar
Select a desired PE from the drop-down list (shown below) on the statusbar in the Main window.

Figure 2.28 Statusbar on Main Window

Caution When the drop-down list used for switching between cores on the status bar is being displayed while the size of this window is maximized, part of the list is hidden behind the task bar and thus cannot be selected.
Set the task bar to "Hide automatically" or set the location of the task bar as [Right], [Left], or [Upper].

(2) Switching through the Debug Manager panel
Select a desired PE on the Debug Manager panel that are opened by selecting [Debug Manager] from the [View] menu.

Figure 2.29 Debug Manager Panel



2.10 Execute Programs

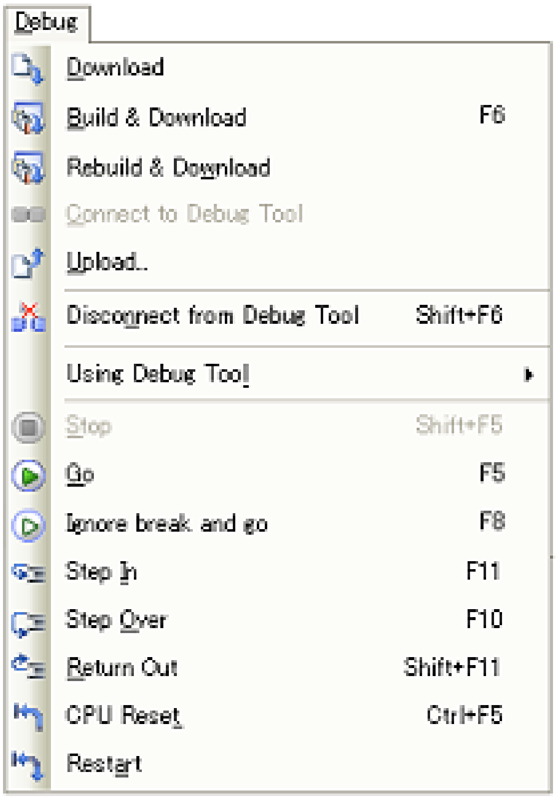
This section describes how to execute programs.
Main operations in this section are taken place from the debug toolbar or the [Debug] menu in the Main window, where commands to control the execution of programs are included.

Caution Items of the debug toolbar and the [Debug] menu are valid only while connected to the debug tool.

Remark For "program execution control" for a microcontroller that supports multi-core, see also to "2.9 Select a Core (PE)".

Figure 2.30 Debug Toolbar (Floating State)

Figure 2.31 [Debug] Menu



2.10.1 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 1. You can automatically overwrite the value of I/O register/CPU register with the specified values after CPU reset under breaking (see "2.20 Use Hook Function" for details).

Remark 2. In cases of failure to reset, the following setting may be required to enable resetting.

- The [Reset While Running] category on the [Debug Tool Settings] tab of the Property panel

[Use the force reset] property	[Yes]

2.10.2 Execute programs

The following types of CS+ execution functions are provided.
Select any of the following operations according to the purpose of debugging.

See "2.11 Stop Programs (Break)" for details on how to stop the program in execution.

2.10.2.1 Execute after resetting microcontroller (CPU)
2.10.2.2 Execute after resetting microcontroller (CPU) (Initial stop debug)
2.10.2.3 Execute from the current address
2.10.2.4 Execute after changing PC value

Remark You can automatically overwrite the value of I/O register/CPU register with the specified values before starting program execution (see "2.20 Use Hook Function" for details).

2.10.2.1 Execute after resetting microcontroller (CPU)

Reset CPU and start 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 "2.11.2 Stop the program manually").
- The PC has reached a breakpoint (see "2.11.3 Stop the program at the arbitrary position (breakpoint)").
- A break event condition has been met (see "2.11.4 Stop the program at the arbitrary position (break event)" or "2.11.5 Stop the program with the access to variables/I/O registers").
- Other break causes have occurred.

Remark 1. This operation is the same as when the button is clicked after clicking the button.

Remark 2. In cases of failure to reset, the following setting may be required to enable resetting.

- The [Reset While Running] category on the [Debug Tool Settings] tab of the Property panel

[Use the force reset] property	[Yes]

2.10.2.2 Execute after resetting microcontroller (CPU) (Initial stop debug)

Reset CPU and start execution of the program from the reset address.
The initial stop state of the CPU can be reproduced because the CPU does not enter the break state after release from the reset state.

Make the following settings before using this function.

- The [Multi-core] category on the [Debug Tool Settings] tab of the Property panel

[Debug initial stop state] property	[Yes]

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 "2.11.2 Stop the program manually").
- The PC has reached a breakpoint (see "2.11.3 Stop the program at the arbitrary position (breakpoint)").
- A break event condition has been met (see "2.11.4 Stop the program at the arbitrary position (break event)" or "2.11.5 Stop the program with the access to variables/I/O registers").
- Other break causes have occurred.

2.10.2.3 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 "2.11.2 Stop the program manually").
- The PC has reached a breakpoint (see "2.11.3 Stop the program at the arbitrary position (breakpoint)").
- A break event condition has been met (see "2.11.4 Stop the program at the arbitrary position (break event)" or "2.11.5 Stop the program with the access to variables/I/O registers").
- Other break causes have occurred.

(2) Execution ignoring break-related events
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 "2.11.2 Stop the program manually").
- Other break causes have occurred.

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

(3) Execution to the caret position
To start this operation, move the caret to the line/instruction 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 "2.11.2 Stop the program manually").
- Other break causes 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 Action event will also be ignored.

2.10.2.4 Execute after changing PC value

The program is executed after forcibly changing the current PC value to an arbitrary address.
To start this operation, move the caret to the line/instruction 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 "2.10.2.3 Execute from the current address".

2.10.3 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 depends on the setting as follows:

- When the button on the Editor panel's 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 Editor panel's toolbar is valid:
  Step execution is conducted in instruction level.

Caution The button is only enabled if the Mixed display mode is selected on the Editor panel.

Step execution is divided into the following types:

Caution 1. Breakpoints, break events, and action events that have been set do not occur during step execution.

Caution 2. An error message will appear while processing a function prologue or epilogue if the return address cannot be acquired.

Caution 3. [Full-spec emulator][E1][E20]

- Interrupts are not acknowledged during step execution.
- It will not go into standby mode during step execution.

Caution 4. [Simulator]
You may jump to an interrupt handler during step execution.

Caution 5. 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 making either or both of the following settings on the [Compile Options] tab in the Property panel of the build tool.

- Set the [Enhance debug information with optimization] property to [Yes(-g_line)] in the [Debug Information] category.
- Set the [Level of optimization] property to [Debug precedence(-Onothing)] in the [Optimization] category.

2.10.3.1 Step in function (Step in execution)

When the function is called, the program is stopped at the top of the called function.
Click the  button on the debug toolbar to perform Step in execution.

Caution 1. Step in execution for a function without the debug information is not possible.

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. The beginning of the function (prologue processing) is not skipped. To skip prologue processing, perform Step in execution again.

2.10.3.2 Step over function (Step over execution)

In the case of a function call by the jarl instruction, all the source lines/instructions in the function are treated as one step and executed until the position where execution returns from the function (step execution will continue until the same nest is formed as when the jarl instruction has been executed).

Click the button on the debug toolbar to perform Step over execution.

In the case of an instruction other than jarl, operation is the same as when the button is clicked.

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

2.10.3.3 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 caller function. When the execution of source line/instruction that require checking has been completed, you can perform step execution using this instruction so that you can make the program return to the caller function without step executing the remaining instructions inside the function.

Click the button on the debug toolbar to perform Return out execution.

Caution 1. If Return out execution is performed in the main function, the program is stopped in the startup routine.

Caution 2. Return out execution cannot be performed immediately after stepping in a function.

Caution 3. Return out execution cannot be performed while processing a function prologue or epilogue.

Caution 4. If Return out execution is performed in a function that called the longjmp function, breaks may not occur.

Caution 5. If Return out execution is performed in a recursive function, the program will be executed in free-run mode.

2.11 Stop Programs (Break)

This section describes how to stop the program in execution.
CS+ can stop the program in execution at the arbitrary position by using the following functions.

(1) Forced break function
Stops the program forcibly.

(2) Hardware break function
The debug tool consecutively checks the break condition while the program is in execution and stops the program when the condition is met. This function is implemented using the debug tool resources.
There are two types of Hardware Break event: "execution type" which stops the program at the arbitrary position; and "access type" which stops the program when an arbitrary variable and so on is accessed with the specified type.

Remark 1. [E1][E20][Full-spec emulator]
Hardware break events (execution type) are of two types: pre-execution breaks, where the break occurs before execution of the instruction at the specified address and post-execution breaks, where the break occurs after execution of the instruction. In CS+, when hardware break events are specified, the resources of the pre-execution breaks are the first to be used. If those resources have been used up, the resources of the post-execution breaks are used (see "2.19.7.1 Restrictions on the numbers of valid events and channels").

Remark 2. [Simulator]
When hardware break events (execution type) are specified, the breaks in the program occur before execution of the instructions at the specified addresses (pre-execution breaks).

(3) Software break function [Full-spec emulator][E1][E20]
Temporarily replaces the instruction code for a specified address with a break instruction and stops the program when this instruction is executed.
If a Software Break event is set, the program will break before executing instruction at the specified address ("before execution" break).

Caution Since an instruction code is replaced by the break instruction, setting or deleting a software break event is followed by programming of the memory at the timing described below.

- When the program execution is started (including the start of execution via [Ignore break and go] from the [Debug] menu)
- When the debug tool is disconnected from CS+

Caution 1. If a forced break is performed while in standby mode (HALT/STOP/IDLE), the current PC position will indicate the address of the next instruction after the standby mode instruction.
This behavior differs depending on the debug tool used.

- [Full-spec emulator][E1][E20]
  The forced break will release standby mode.
- [Simulator]
  The forced break will not release standby mode.
  It will appear that standby mode has been released. Check the CPU status on the Main window's statusbar to see if standby mode has been released.

Caution 2. [Full-spec emulator][E1][E20]
Do not decrease the voltage of the target system during a break. A reset that is generated by the low-voltage detector (LVI) or by power-on-clear (POC) during a break causes an incorrect operation of CS+ or communication errors.
A break during emulation of power supply off also causes communication errors.

Remark 1. For "program execution control" or "event occurrence" for a microcontroller that supports multi-core, see also to "Select a Core (PE)".

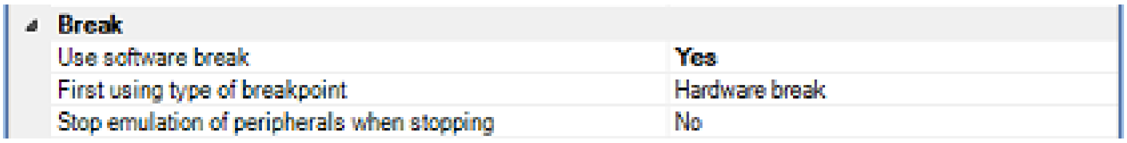
Remark 2. When the program in execution is stopped, a statement of the cause of the break appears on the Statusbar in the Main window.

2.11.1 Configure the break function [Full-spec emulator][E1][E20]

Before the break function can be used, it is necessary to make settings relating to the operation of a break.
This break operation can be configured in the [Break] category on the [Debug Tool Settings] tab of the Property panel.

Remark **[Simulator]**
The settings relating to the operation of a break are not necessary.

Figure 2.32 [Break] Category [Full-spec emulator][E1][E20]



(1) [Use software break]
Select whether to use the Software break function [Full-spec emulator][E1][E20].
Select [Yes] to use the software break function (default: [No]).

Caution 1. If this property is set to [No] after you have used the software break function, all software break events and Printf events that have been set will be disabled. Selecting [Yes] in this state does not automatically restore the events, so you will need to manually enable them.

Caution 2. This property cannot be changed during program execution.

(2) [First using type of breakpoint]
This property appears only when the [Use software break] property is set to [Yes].
Select from the following drop-down list the type of a breakpoint to use with priority when setting it with a one click operation of the mouse in the Editor panel/Disassemble panel.

Hardware break	Sets hardware breakpoint with priority, by using the Hardware break function (default). Once set, it is treated as a Hardware Break event (execution system).
Software break	Sets software breakpoint with priority, by using the Software break function [Full-spec emulator][E1][E20]. Once set, it is treated as a Software Break event.

Caution If the number of the set breakpoints of the specified type exceeds the limit settable (see "2.19.7.1 Restrictions on the numbers of valid events and channels"), a breakpoint of another type will be used.

(3) [Stop emulation of peripherals when stopping]
Select whether to terminate the peripheral emulation while stopping the program execution (Peripheral Break).
Select [Yes] to terminate (default: [No]).

2.11.2 Stop the program manually

The program in execution is forcibly stopped by clicking the button on the debug toolbar (Forced break function).

Remark In cases of failure to stop a program during execution, the following setting may be required to allow stopping execution of the program.
However, stopping a program while this setting is in place leads to a CPU reset.

- The [Reset While Running] category on the [Debug Tool Settings] tab of the Property panel

[Use the force reset] property	[Yes]

2.11.3 Stop the program at the arbitrary position (breakpoint)

A breakpoint is one of the break events that can be set by one-clicking with the mouse.
The program in execution can be stopped at the arbitrary position easily by setting a breakpoint.
This section describes the following operations.

2.11.3.1 Set a breakpoint
2.11.3.2 Edit a breakpoint
2.11.3.3 Delete a breakpoint

2.11.3.1 Set a breakpoint

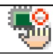
Breakpoints can be set via the Editor panel/Disassemble panel in which the source text/disassembled text is displayed.

Within the Main area (Editor panel) or Event area (Disassemble panel) in which a valid address is displayed, click on the location where you want to set a breakpoint. A breakpoint whose type is being selected in the [First using type of breakpoint] property 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 (Hardware Break or Software Break) has been set at the target address, and it is managed in the Events panel (see "2.19 Manage Events" for details).

Table 2.6 Event Marks of Breakpoint

Type of Breakpoint	Event Type	Event Mark
Hardware breakpoint	Hardware Break event[Note]	
Software breakpoint [Full-spec emulator][E1][E20]	Software Break event[Note]	

Note In the [Name] area of the Events panel, "Break" is displayed as the event type name.

Figure 2.33 Breakpoint Setting Example (Disassemble Panel)

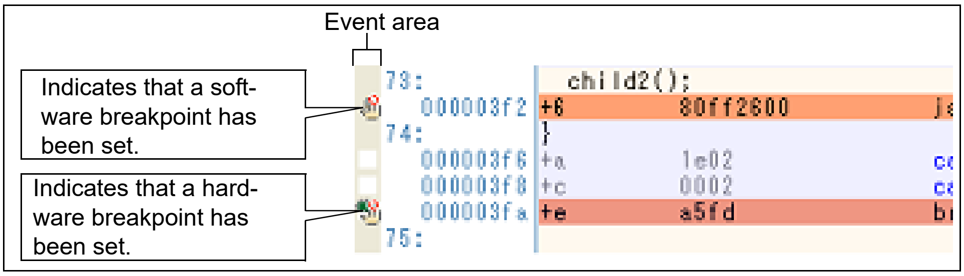
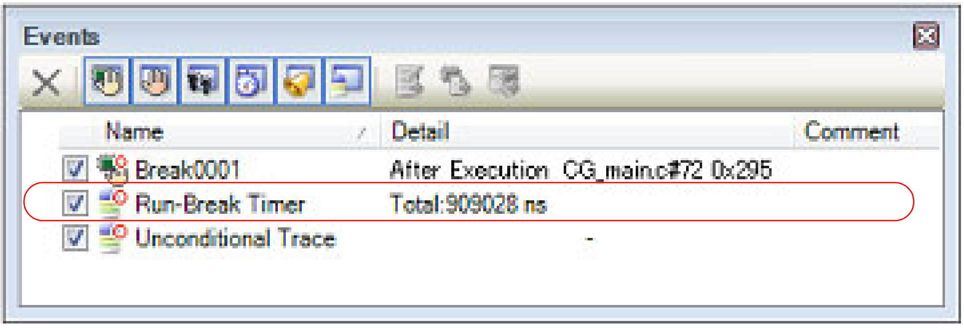


Figure 2.34 Example of Setting Breakpoint in Events Panel



Caution 1. Since a breakpoint is set as a break event and managed as a event, restrictions apply to the number of breakpoints that can be simultaneously set. Also see "2.19.7 Notes for setting events" for details on breakpoints (e.g. limits on the number of enabled events).

Caution 2. Breakpoints can only be set at lines that have valid addresses.

Caution 3. [Full-spec emulator][E1][E20]
Software breakpoints can be set in the code flash area or internal RAM area.
If you wish to set software breakpoints in an internal RAM area, select [Yes] for the [Initialize RAM when connecting] property in the [Connection with Target Board] category on the [Connect Settings] tab of the Property panel.

Remark 1. Event marks differ depending on the event state (see "2.19.1 Change the state of set events (valid/invalid)").
When an event is set at the point where other event is already set, the event mark () is displayed meaning more than one event is set at the point.

Remark 2. **[Full-spec emulator][E1][E20]**
You can set hardware breakpoints/software breakpoints without depending on the selection of the [First using type of breakpoint] property by the operation described below.
Note, however, that "Operation1" is only available in the Disassemble panel.

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.

Remark 3. **[Simulator]**
The type of breakpoint that can be set is locked to hardware breakpoints.

2.11.3.2 Edit a breakpoint

It is possible to edit a breakpoint you have set.
For details on how to do it, see "2.19.4.1 Edit execution-related events".

Caution This function applies to only a breakpoint whose type is Hardware Break.

2.11.3.3 Delete a breakpoint

Click event marks displayed in the Editor panel/Disassemble panel to delete set breakpoints (the event mark will be erased).

2.11.4 Stop the program at the arbitrary position (break event)

The program in execution can be stopped at the arbitrary position by setting a break event (execution type).
This section describes the following operations.

2.11.4.1 Set a break event (execution type)
2.11.4.2 Edit a break event (execution type)
2.11.4.3 Delete a break event (execution type)

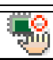
2.11.4.1 Set a break event (execution type)

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, in accordance with your desired event type, after moving the caret to the target line that has a valid address.

Event Type	Operation	Description
Hardware Break	Select [Break Settings] >> [Set Hardware Break]	Sets a Hardware Break event by using the Hardware break function.
Software Break [Full-spec emulator][E1][E20]	Select [Break Settings] >> [Set Software Break]	Sets a Software Break event by using the Software break function [Full-spec emulator][E1][E20].

A break event is set to the instruction at the start address corresponding to the line of the caret position.
When a break event (execution type) is set, the following event mark is displayed in the event area of the line that an event is set., and the source text line or disassembled text line will be highlighted.
When you have performed this operation, the set break event is managed in the Events panel as a Hardware Break event (execution type)/Software Break event (execution type) (see "2.19 Manage Events" for details).

Table 2.7 Event Marks of Break Event (Execution Type)

Event Type	Event Mark
Hardware Break	

Event Type	Event Mark
Software Break [Full-spec emulator][E1][E20]	

Figure 2.35 Break Event (Execution Type) Setting Example (Disassemble Panel)

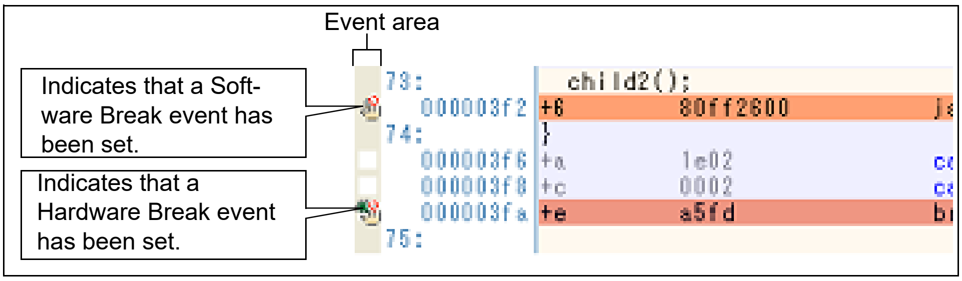
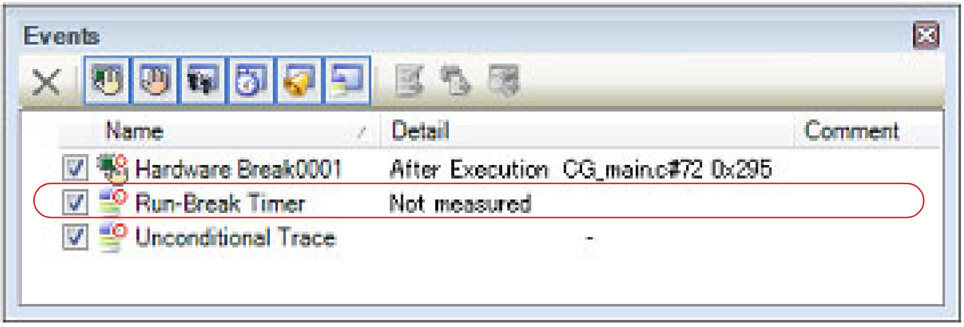


Figure 2.36 Example of Setting Hardware Break Event (Execution Type) in Events Panel



Caution 1. When setting a break event (execution type), also see "2.19.7 Notes for setting events" for details (e.g. limits on the number of valid events).

Caution 2. [Full-spec emulator][E1][E20]
Software breakpoints can be set in the code flash area or internal RAM area.
If you wish to set software breakpoints in an internal RAM area, select [Yes] for the [Initialize RAM when connecting] property in the [Connection with Target Board] category on the [Connect Settings] tab of the Property panel.

Remark Event marks differ depending on the event state (see "2.19.1 Change the state of set events (valid/invalid)"). When an event is set at the point where other event is already set, the event mark ( ) is displayed meaning more than one event is set at the point.

2.11.4.2 Edit a break event (execution type)

It is possible to edit a break event (execution type) you have set.
For details on how to do it, see "2.19.4.1 Edit execution-related events".

Caution This function applies to only a break event (execution type) whose type is Hardware Break.

2.11.4.3 Delete a break event (execution type)

To delete a break event (execution type) you have set, click the event mark displayed in the Editor panel/Disassemble panel.

Also, there is another way to delete a set break event. Select a Software Break event/Hardware Break event in the Events panel, and then click the button in the toolbar (see "2.19.5 Delete events").

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

By setting a break event with the access, the program can be stopped when an arbitrary variable or I/O register is accessed with the specified type.

You can also limit the accessed value.

The following types can be specified with the access.

Table 2.8 Types of Accesses to Variables

Access Type	Description
Read	The program is stopped with the read access to (after reading) the specified variable/I/O register.
Write	The program is stopped with the write access to (after writing) the specified variable/I/O register.
Read/Write	The program is stopped with the read access/write access to (after reading or writing) the specified variable/I/O register.

Caution The program is not stopped with the access via DMA (Direct Memory Access).

This section describes the following.

2.11.5.1 Set a break event (access type)

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

Caution Also see "2.19.7 Notes for setting events" for details on breakpoints (e.g. limits on the number of enabled events).

(1) Set a break event to a variable/I/O register in the Editor panel/Disassemble panel
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, in accordance with your desired access type, after selecting an arbitrary variable or I/O register on the source text or the disassembled text. Note, however, that only global variables, static variables inside functions, and file-internal static variables can be used.

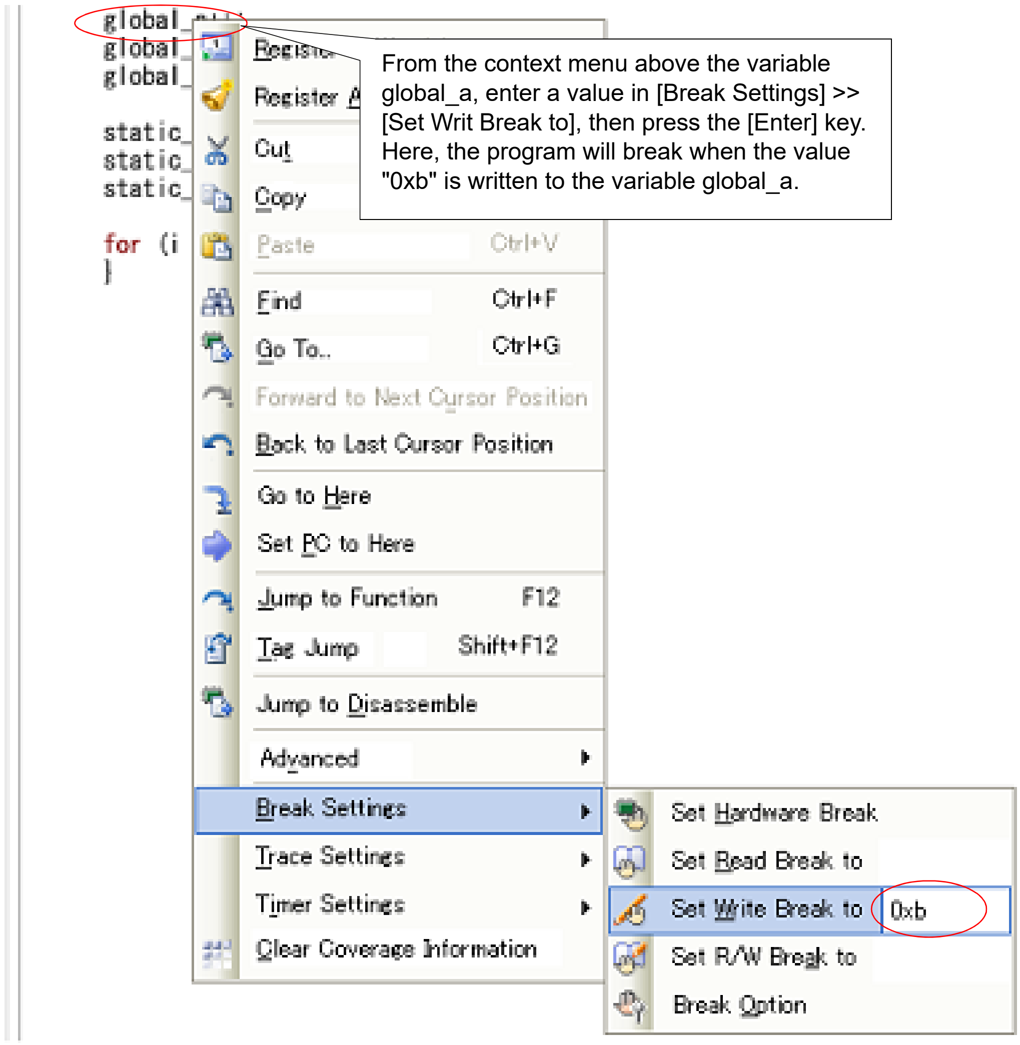
Access Type	Operation
Read	Select [Break Settings] >> [Set Read Break to], and then press the [Enter] key.
Write	Select [Break Settings] >> [Set Write Break to], and then press the [Enter] key.
Read/Write	Select [Break Settings] >> [Set R/W Break to], and then press the [Enter] key.

At this time, if you have specified a value in the text box in the context menu, break will occur only when the specified value is used for the reading, writing or reading/writing. On the other hand, if no value is specified, reading., writing or reading/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 2.37 Example of Setting Break Event (Access Type) on Variable in Editor Panel



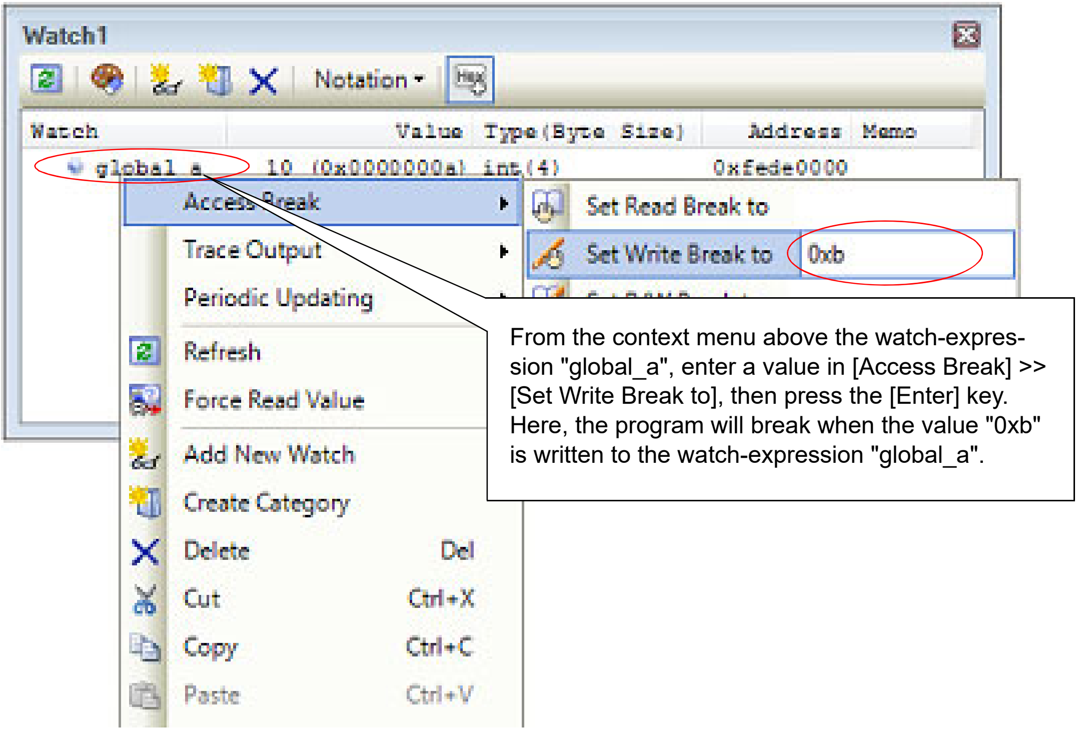
(2) Set a break event (access type) to a registered watch-expression
You can set break events in the Watch panel.
Follow the operation listed below from the context menu, in accordance with your desired access type, after selecting the registered watch-expression (multiple selections not allowed).
Note, however, that only global variables, static variables inside functions, file-internal static variables, and I/O register can be used.

Access Type	Operation
Read	Select [Access Break] >> [Set Read Break to], and then press the [Enter] key.
Write	Select [Access Break] >> [Set Write Break to], and then press the [Enter] key.
Read/Write	Select [Access Break] >> [Set R/W Break to], and then press the [Enter] key.

At this time, if you have specified a value in the text box in the context menu, break will occur only when the specified value is used for the reading., writing or reading/writing. On the other hand, if no value is specified, reading., writing or reading/writing the selected watch-expression by any value will cause the break to occur.

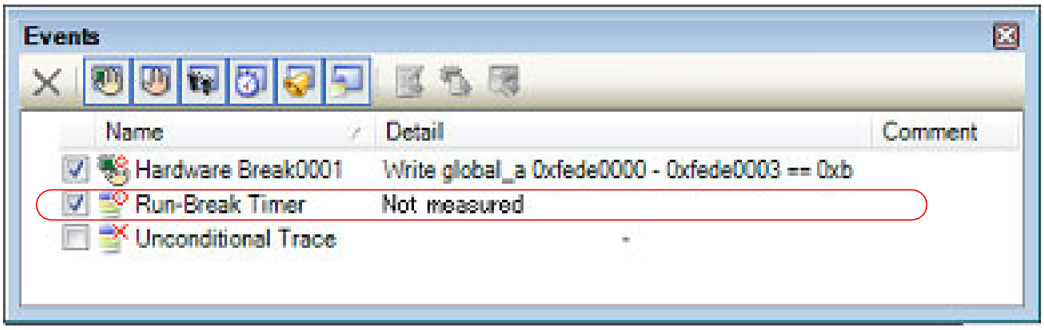
Caution A watch-expression within the current scope can be specified.
To target a watch-expression outside the current scope, select a watch-expression with a specified scope.

Figure 2.38 Example of Setting Hardware Break Event (Access Type) on Watch-Expression



When you have performed the above operation, the set break event (access type) is managed in the Events panel as a Hardware Break event (access type) (see "2.19 Manage Events" for details).

Figure 2.39 Example of Setting Hardware Break Event (Access Type) in Events Panel



2.11.5.2 Edit a break event (access type)

It is possible to edit a break event (access type) you have set.
For details on how to do it, see "2.19.4.2 Edit access-related events".

2.11.5.3 Delete a break event (access type)

To delete a break event (access type) you have set, select a Hardware Break event in the Events panel, and then click the button in the toolbar (see "2.19.5 Delete events").

2.11.6 Other break causes

The cause of the break other than the described above is as follows:
You can confirm the break cause with the Status message on the statusbar in the Main window, Output panel, or Trace panel **[Simulator]**.

Table 2.9 Other Break Causes

Break Cause	Debug Tool			
	Full-spec emulator	E1/E20	E2	Simulator
Full of the trace memory[Note 1]	✓	✓	✓	✓
An access to non-mapped area	-	-	-	✓
A writing to write-protected area	-	-	-	✓
An occurrence of Temporary Break[Note 2]	✓	✓	✓	✓
Step execution count-over	✓	✓	✓	✓
An occurrence of Relay Break[Note 3]	✓	✓	✓	✓
E2 expansion function	-	-	✓	-
Fully used the storage memory	-	-	✓	-
Fully used the storage memory at LPD output of software trace	-	-	✓	-

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. A break that is internally used by CS+ (Users cannot use it.)

Note 3. A break for a synchronous execution and a synchronous break, in a microcontroller that supports multi-core

2.12 Display/Change the Memory, Register and Variable

This section describes how to display/change the memory, register and variable.

Remark For "Information of the memory, registers, or variables" for a microcontroller that supports multi-core, see also to "Select a Core (PE)".

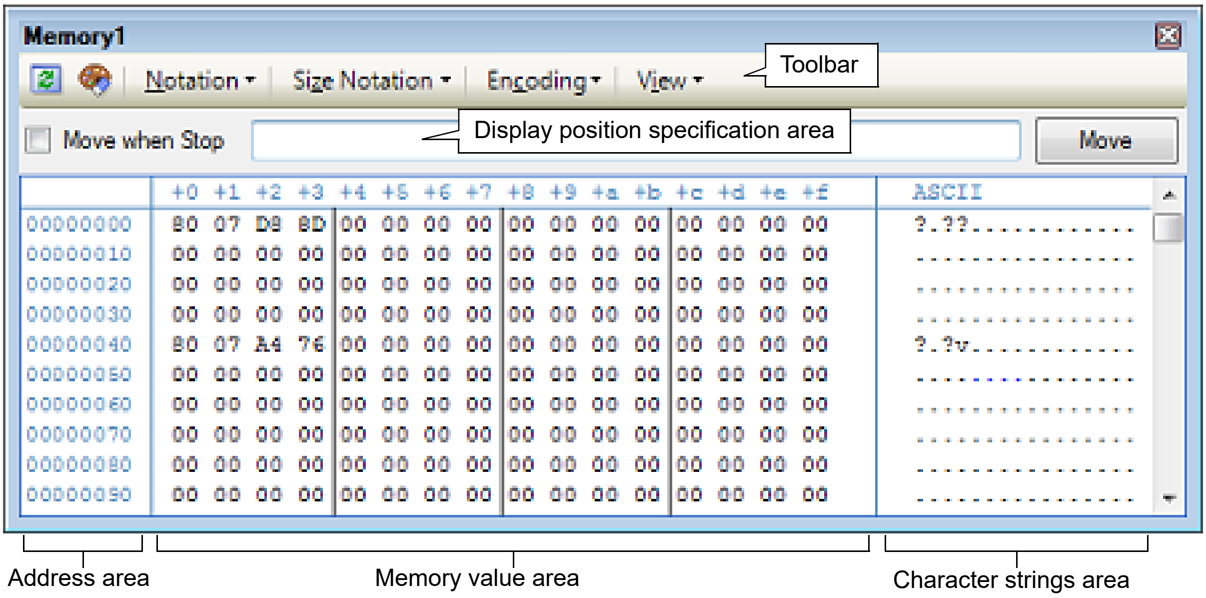
2.12.1 Display/change the memory

The contents of the memory can be displayed and its values can be changed in the Memory panel below.

Select the [View] menu >> [Memory] >> [Memory*1 - 4*]. The maximum of 4 Memory panels can be opened. Each panel is identified by the names "Memory1", "Memory2", "Memory3" and "Memory4" on the titlebar.

For details on the contents and function in each area, see the section for the Memory panel.

Figure 2.40 Display Memory Contents



Remark You can set the scroll range (as start and end address) of the vertical scroll bar on this panel via the Scroll Range Settings dialog box which is opened by clicking the button from [View] on the toolbar.

This section describes the following.

2.12.1.1 Specify the display position

It is possible to specify the display start position of the memory contents by specifying an address expression in the display position specification area (starting with address 0x0 by default).

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 2.41 Display Position Specification Area (Memory Panel)

(1) Specify an address expression
Directly enter the address expression of the memory value address to display in the text box. You can specify an input expression with up to 1024 characters. The result of the expression is treated as the display start position address.
Note that address values greater than the microcontroller address space cannot be specified.

Remark 1. A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (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) Specify automatic/manual evaluation of the address expression
The timing to change the display start position can be determined by specifying in the [Move when Stop] check box and the [Move] button.

[Move when Stop]	☑	The caret is moved to the address which is automatically calculated from the address expression after the program is stopped.
	☐	The address expression is not automatically evaluated after the program is stopped. Click the [Move] button to manually evaluate the address expression.
[Move]		When the [Move when Stop] check box is not checked, click this button to evaluate the address expression and move the caret to the result address of the evaluation.

2.12.1.2 Change display format 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	The following buttons to change the notation of memory values are displayed.
Hexadecimal	Displays memory values in hexadecimal number (default).
Signed Decimal	Displays memory values in signed decimal number.
Unsigned Decimal	Displays memory values in unsigned decimal number.
Octal	Displays memory values in octal number.
Binary	Displays memory values in binary number.
Size Notation	The following buttons to change the notation of sizes 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.

Encoding	The following buttons to change the encoding of 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 Big-Endian	Displays character strings in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character strings in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character strings in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character strings in UTF-32 Little-Endian 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[Note].
Float Complex	Displays character strings as a complex number of single-precision floating-point[Note].
Double Complex	Displays character strings as a complex number of double-precision floating-point[Note].
Float Imaginary	Displays character strings as an imaginary number of single-precision floating-point[Note].
Double Imaginary	Displays character strings as an imaginary number of double-precision floating-point[Note].
View	The following buttons to change the display format are displayed.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range for this panel.
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.

2.12.1.3 Modify the memory contents

The memory values can be edited.
Directly edit from the keyboard after moving the caret to the line to modify in memory value area/characters area.
The color of the memory value changes when it is in editing. Press the [Enter] key to write the edited value to the target memory (if the [Esc] key is pressed before the [Enter] key is pressed, the editing is cancelled).
However, the character string that can be inputted during the editing is limited to that character string that can be handled by the display notation that has been currently specified. In the character strings area, modification can only be made with "ASCII" character code.
This operation can be taken place while the program is in execution. See "2.12.1.4 Display/modify the memory contents during program execution" for details on how to operate it.

When you modify the values, be aware of the following examples.

Example 1. The value exceeds the upper limit of the display bit wide
If you edit the display value "105" as "1" to "3" in the decimal 8-bit display, the value will be changed to the upper limit of "127".

Example 2. The symbol, "-" is entered between numbers
If you edit the display value "32768" as "32-68" with signed decimal 16-bit display, "3" and "2" are changed to the blank and the value is changed to "-68".

Example 3. The blank symbol (space) is entered between numbers
If you edit the display value "32767" as "32 67", "3" and "2" are changed to the blank and the value is changed to "67".

Example 4. The same value is entered
Even if the same value as the current memory value is specified, the specified value is written to the memory.

2.12.1.4 Display/modify the memory contents during program execution

The Memory panel/Watch panel has the real-time display update function that can update/modify the display contents of the memory/watch-expression in real-time while executing the program.
Using the real-time display update function allows you to display/modify the value of memory/watch-expression not only while the program is stopped, but also in execution.
The real-time display update function is realized by the CPU's/debug tool's RRM function (reading) [Simulator], RAM monitor function (reading) [Full-spec emulator][E1][E20] and DMM function (modifying). Each function has a different area that can be used for reading and writing.
Enable the real-time display update function by making the basic settings below on the [Debug Tool Settings] tab of the Property panel.

Table 2.10 Basic Settings for Real-time Display Update Function

Category	Property	Set Value
[Access memory while running]	[Update display during the execution]	[Yes] (default)
	[Display update interval[ms]]	[*Integer number between 100 and 65500*]

Caution 1. Local variables are not subject to the real-time display update function.

Caution 2. When a 2-, 4-, or 8-byte variable is to be read through the RRM or RAM monitor function, the process of assigning a value to the variable may be divided into two steps.
If reading of the variable takes place between the two steps, an incorrect value may be read out because the assignment is not completed.

Caution 3. The contents of memory or watch expressions can be read for access in all PEs when the selected microcontroller supports multi-core. In the Local RAM self area, note that they can be read only for the access in the currently selected PE.

Remark See "2.12.1.3 Modify the memory contents" or "2.12.6.6 Modify the contents of watch-expressions" for details on how to modify values in the Memory panel/Watch panel.

(1) RRM function (reading) [Simulator]
This function is used to read the contents of memory or of watch-expressions in real-time during execution of a program.
The following area can be read by the RRM function. Memory and watch-expressions allocated to this area can always be displayed in real-time.

Table 2.11 Target Area of RMM Function

Area	Simulator
Internal ROM	✓
Internal RAM	✓
Peripheral I/O area	-
Data flash	-
Emulation memory	-
Target memory	-

Area	Simulator
CPU register	✓ Note
I/O register (except read-protected I/O registers)	✓

Note Impossible during tracer/timer execution

(2) RAM monitor function (reading) [Full-spec emulator][E1][E20]
This function is used to read the contents of memory or a watch-expression via the CPU's RAM monitor function.
The following area can be read by the RAM monitor function.

Caution If CPU status shifts to the standby mode (HALT/STOP/IDLE) mode, a monitor time-out error will occur.

Table 2.12 Target Area of RAM Monitor Function

Area	Full-spec emulator	E1/E20
Internal ROM	-	-
Internal RAM	✓	✓
Peripheral I/O area	-	-
Data flash (except ID tag)	-	-
Target memory	-	-
CPU register	-	-
I/O register	-	✓ Note

Note Possible only for the RH850G4MH, RH850G4KH

Note that to enable the RAM monitor function, the setting below is required in addition to the Basic Settings for Real-time Display Update Function.

Debug Tool	Category	Property	Set Value
Full-spec emulator E1/E20	[Access memory while running]	[Access during the execution]	[Yes]

(3) DMM function (modifying)
This function is used to write to the memory or watch-expressions in real-time during execution of a program.
The following area can be modified by the DMM function.

Caution 1. If a value is written through the DMM function, the atomicity cannot be guaranteed.

Caution 2. If CPU status shifts to the standby mode (HALT/STOP/IDLE) mode, a monitor time-out error will occur.

Table 2.13 Target Area of DMM Function

Area	Full-spec emulator	E1/E20	Simulator
Internal ROM	-	-	✓
Internal RAM	✓	✓	✓
Peripheral I/O area	-	-	-
Emulation memory	-	-	-
Target memory	-	-	-
CPU register	-	-	✓ Note 1

Area	Full-spec emulator	E1/E20	Simulator
I/O register (except read-protected I/O registers)	-	✓ Note 2	✓

Note 1. Impossible during tracer/timer execution

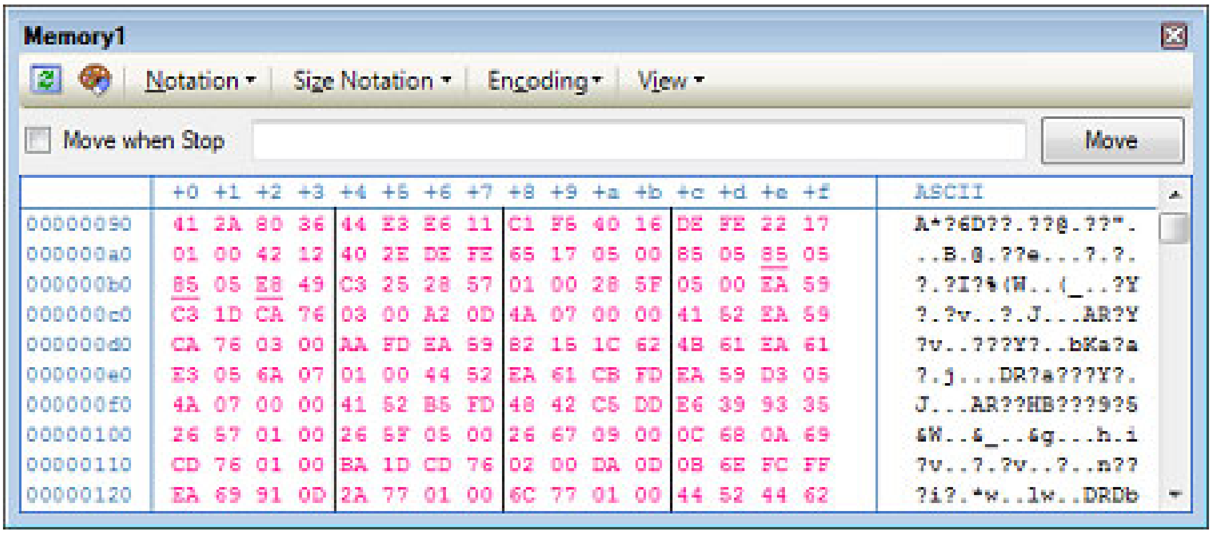
Note 2. Possible only for the RH850G4MH, RH850G4KH

To enable the DMM function, the setting below is required in addition to the Basic Settings for Real-time Display Update Function.

Debug Tool	Category	Property	Set Value
Full-spec emulator E1/E20	[Access memory while running]	[Access during the execution]	[Yes]
Simulator	No setting is required.		

The memory values/watch-expressions updated by the real-time display update function are highlighted in pink on the Memory panel/Watch panel.

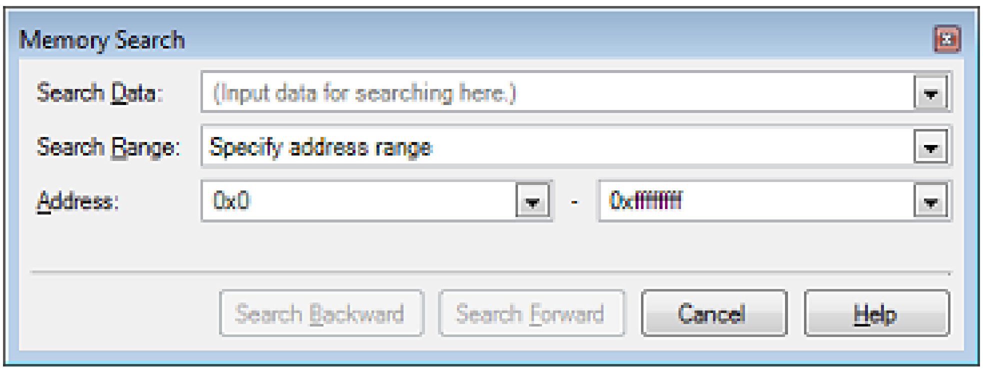
Figure 2.42 Example of Memory Display by Real-time Display Update Function



2.12.1.5 Search the memory contents

Values of memory can be searched in the Memory Search dialog box that is opened by selecting [Find...] from the context menu. The search is operated either in the memory value area or character strings area, in which the caret exists. In this dialog box, follow the steps below.

Figure 2.43 Search Memory Contents (Memory Search Dialog Box)



Caution 1. The contents of the memory cannot be searched during execution of a program.

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

(1) Specify [Search Data]
Specify data to search.
You can either type a value directly into the text box (up to 256 bytes), or select one from the input history via the drop-down list (up to 10 items).
If the search is performed in the memory value area, the value must be entered in the same display format (notation and size) as that area.
If the search is performed in the character strings area, then the target of the search must be a string. The specified string is converted into the encoding format displayed in that area, and searched for.
If a memory value was selected immediately prior to opening this dialog box, then that value will appear as default.

(2) Specify [Search Range]
Select the range to search from the following drop-down list.

Specify address range	Searches in the address range specified in the [Address] area.
Memory mapping	Searches within the selected memory mapping range. This list item displays individual memory mapping configured in the Memory Mapping dialog box. Display format: <*memory type*> <*address range*> <*size*>

(3) Specify [Address]
This item is only enabled if [Specify address range] is selected in the (2) Specify [Search Range].
Specify the range of memory address to search via the start and end addresses. 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).
The results of calculating the address expressions you have entered are treated as start and end addresses, respectively.
Note, however, that the largest address that can be searched is the maximum address of the program space (0x03FFFFFF) (the mirror area cannot be searched).
An address value greater than the value expressed within 32 bits cannot be specified.

Remark 1. A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

Remark 2. If the start address field is left blank, it is treated as if "0x0" were specified.

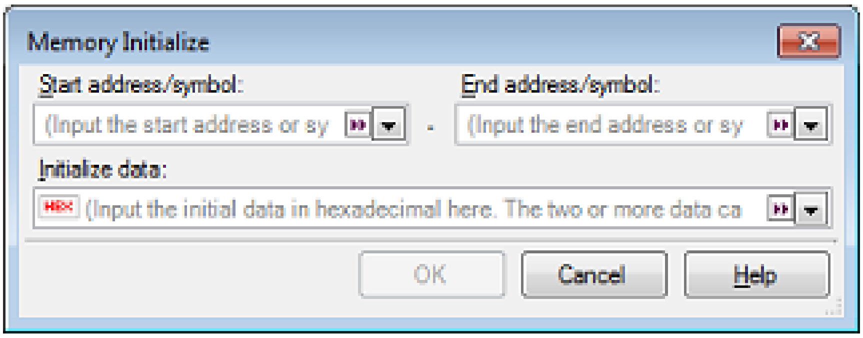
Remark 3. If the end address field is left blank, then it is treated as if the maximum value in the microcontroller's address space were specified.

(4) Click the [Search Backward]/[Search Forward] button
When the [Search Backward] button is clicked, search will start in the order from the large address number to small and the search results are displayed selected in the Memory panel.
When the [Search Forward] button is clicked, search will start in the order from the small address number to small and the search results are displayed selected in the Memory panel.

2.12.1.6 Modify the memory contents in batch (initialize)

Contents of the memory can be modified in batch (initialize).
When [Fill...] from the context menu is selected, the Memory Initialize dialog box opens to modify the memory value of the specified address range in batch.
In this dialog box, follow the steps below.

Figure 2.44 Modify Memory Contents in Batch (Memory Initialize Dialog Box)



(1) Specify [Start address/symbol] and [End address/symbol]
Specify the range of memory address to initialize via the [Start address/symbol] and [End address/symbol]. 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).
The results of calculating the address expressions you have entered are treated as start and end addresses, respectively.
Note that address values greater than the microcontroller address space cannot be specified.

Caution You cannot specify the range of address aligned across the different endian area.

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

(2) Specify [Initialize data]
Specify the initializing data to write to the memory.
You can either type the initial value into the text box directly in hexadecimal number, or select one from the input history via the drop-down list (up to10 items).
You can specify more than one initial value. Specify up to 16 values of up to 4 bytes (8 characters) each, separated by spaces.
Each initial value is parsed from the end of the string, with each two characters interpreted as a byte.
If the string has an odd number of characters, then the first character is interpreted as one byte.
Note that if a initial value consists of more than one byte, then the target memory is overwritten with the value converted into an array of bytes of the specified address range's endian, as follows:

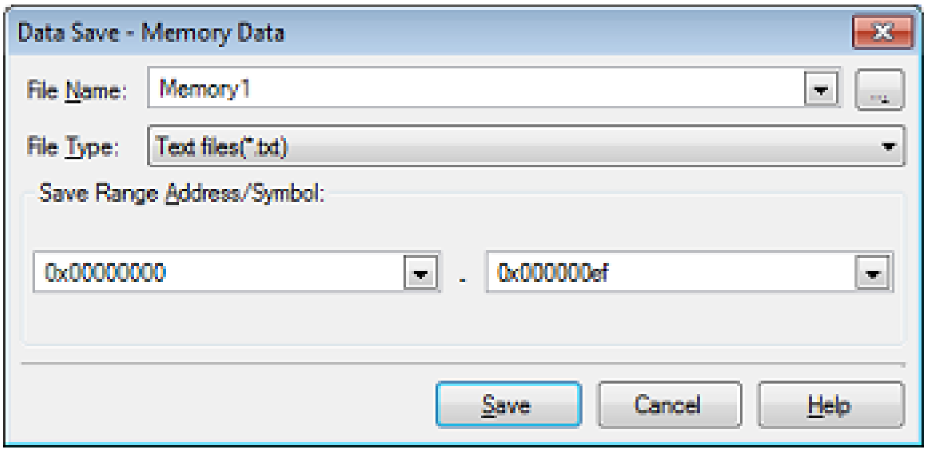
Input Character Strings (Initial Value)	How Data is Overwritten (in Bytes)	
	Little Endian	Big Endian
1	01	01
0 12	00 12	00 12
00 012 345	00 12 00 45 03	00 00 12 03 45
000 12 000345	00 00 12 45 03 00	00 00 12 00 03 45

(3) Click the [OK] button
Click the [OK] button.
The memory area in the specified address range is repeatedly overwritten with the specified initial data pattern. If the end address is reached in the middle of the pattern, then writing ends at that point.
Note that if an illegal value is specified, a message will appear, and the memory value will not be initialized.

2.12.1.7 Save the memory contents

Contents of the memory can be saved with range selection in text files (*.txt)/CSV files (*.csv).
When saving to the file, the latest information is acquired from the debug tool, and it is saved in accordance with the display format on this panel.
The Data Save dialog box can be opened by selecting the [File] menu >> [Save Memory Data As...] (when this operation is taken place with range selection on the panel, the memory data only in the selected range is saved).
In this dialog box, follow the steps below.

Figure 2.45 Save Memory Data (Data Save Dialog Box)



(1) Specify [File Name]
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 [...] button, and selecting a file via the Select Data Save File dialog box.

(2) Specify [File Type]
Select the format in which to save the file from the following drop-down list.
The following file formats can be selected.

List Item	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) Specify [Save Range Address/Symbol]
Specify the range of addresses to save via "start address" and "end addresses".
Directly enter hexadecimal number/address expression in each text box or select from the input history displayed in the drop-down list (up to 10 items).
If a range is selected in the panel, that range is specified as the default. If there is no selection, then the range currently visible in the panel is specified.

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

(4) Click the [Save] button
Saves the memory data to a file with the specified filename, in the specified format.

Figure 2.46 Output Example of Memory Data

[Text files (*.txt)]
(Hexadecimal notation/8-bit width/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 |
```

[CSV files (*.csv)]
(Hexadecimal notation/8-bit width/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,
```

Remark When the contents of the panel are overwritten by selecting the [File] menu>> [Save Memory Data], each Memory panel (Memory1-4) is treated as a different panel.
In addition, saving range is same as the previously specified address range.

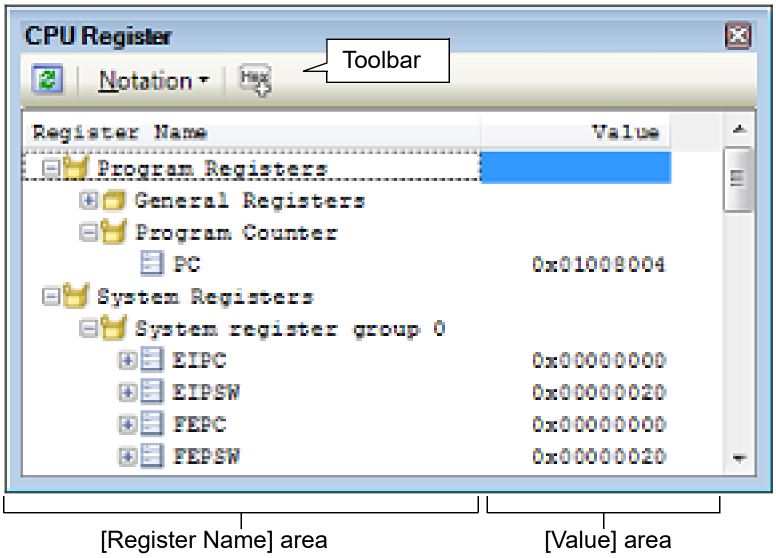
2.12.2 Display/change the CPU register

The contents of the CPU register (program registers/system registers) can be shown and the value can be changed in the CPU Register panel below.

Select the [View] menu >> [CPU Register].

For details on the contents and function in each area, see the section for the CPU Register panel.

Figure 2.47 Display the Contents of CPU Register (CPU Register Panel)



This section describes the following.

2.12.2.1 Change display format of values

The display format of the [value] area can be changed using buttons below on the toolbar.

Notation	The following buttons to change the notation of a data value are displayed.

AutoSelect	Displays the value of the selected item (including sub-items) in the default notation (default).
Hexadecimal	Displays the value of the selected item (including sub-items) in hexadecimal number.
Signed Decimal	Displays the value of the selected item (including sub-items) in signed decimal number.
Unsigned Decimal	Displays the value of the selected item (including sub-items) in unsigned decimal number.
Octal	Displays the value of the selected item (including sub-items) in octal number.
Binary	Displays the value of the selected item (including sub-items) in binary number.
ASCII	Displays the character strings of the selected item (including sub-items) in ASCII code. If the character size is 2 bytes and above, it is displayed with the characters for each 1 byte arranged side-by-side.
Float	Displays the value of the selected item in float. Note that when the value is not 4-byte data, displays it in the default notation.
Double	Displays the value of the selected item in double. Note that when the value is not 8-byte data, displays it in the default notation.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value.

2.12.2.2 Modify the CPU register contents

The CPU register values can be edited.
Select the value of the CPU register to edit in the [Value] area, then click on it again to switch the value to edit mode (press the [Esc] key to cancel the edit mode).
To write the edited value to the target memory, directly enter the value from the keyboard then press the [Enter] key.

Caution This operation cannot be performed during program execution.

2.12.2.3 Display/modify the CPU register contents during program execution

By registering a CPU register to the Watch panel as a watch-expression, the value of the CPU register can be displayed/modified not only while the program is stopped, but in execution.
See "2.12.6 Display/change watch-expressions" for details on the watch-expression.

2.12.2.4 Save the CPU register contents

The Save As dialog box can be opened by selecting the [File] menu >> [Save CPU Register Data As...], and all the contents in the CPU register can be saved to a text file (*.txt) or CSV file (*.csv).
When saving to files, retrieve the latest information from the debug tool.

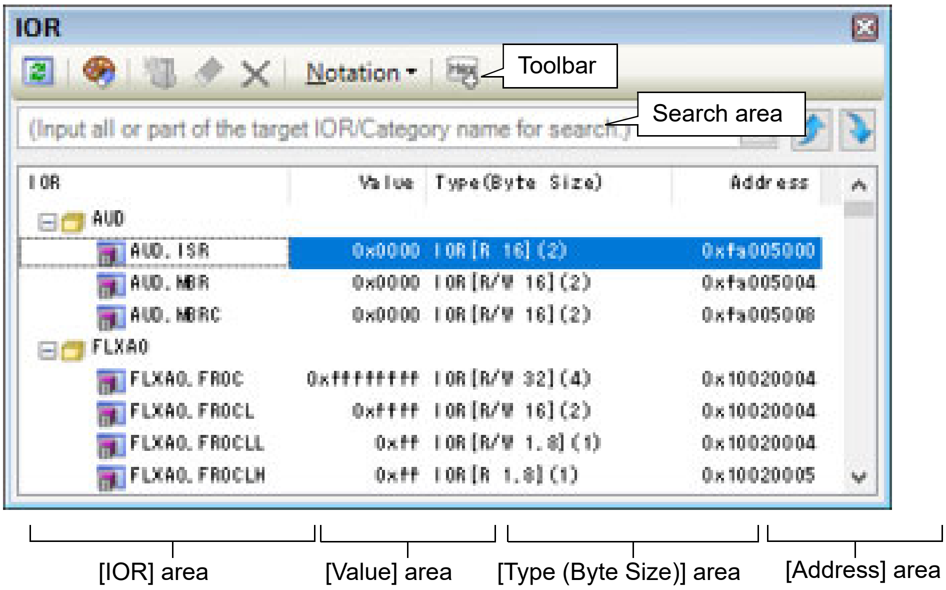
Figure 2.48 Output Example of CPU Register Data

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

2.12.3 Display/change the I/O register

Contents of the I/O register can be displayed and its values can be changed in the IOR panel below.
Select the [View] menu >> [IOR].
For details on the contents and function in each area, see the section for the IOR panel.

Figure 2.49 Display the Contents of I/O register (IOR Panel)



This section describes the following.

2.12.3.1 Search for an I/O register

An I/O register can be searched for.

Specify the I/O register name to search with the text box in the search area (case-insensitive). You can either type character strings directly from the key board (up to 512 characters), or select one from the input history via the drop-down list (up to 10 items). Then, click either one of the following button.

	Searches up for the I/O register name containing the string specified in the text box, and selects the I/O register that is found.
	Searches down for the I/O register name containing the string specified in the text box, and selects the I/O register that is found.

Remark 1. The hidden I/O register name being classified with a category can be searched (the category is opened and the I/O register is selected).

Remark 2. After typing character strings to search, to press the [Enter] key is the same function as clicking the button, and to press the [Shift] + [Enter] key is the same function as clicking the button.

2.12.3.2 Organize I/O registers

The each I/O register can be categorized (by folders) and displayed in the tree view.

Caution 1. Categories cannot be created within categories.

Caution 2. I/O registers cannot be added or deleted.

(1) Create new category
Move the caret to the I/O register name to create a new category then click the button in the toolbar and directly enter the new category name.

(2) Edit category name
Click the category name to edit, and click it again, then directly modify the category name from the keyboard.

(3) Delete categories
Select categories to delete then click the button in the toolbar.
However, the categories that can be deleted are only the empty categories.

(4) Change the display order
I/O register name is categorized when I/O register is dragged and dropped in the category.
Also, the display order of the categories and the I/O register names (upper or lower position) can be changed easily by drag and drop operation.

(5) Restore the category to its initial state
click the button in the toolbar. The edited category is cleared and displayed with the initial category classification.

2.12.3.3 Change display format of values

The display format of the [value] area can be changed using buttons below on the toolbar.

Notation	The following buttons to change the notation of a data value are displayed.
Hexadecimal	Displays the value of the selected item in hexadecimal number (default).
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned Decimal	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item.

2.12.3.4 Modify the I/O register contents

The I/O register values can be edited.
Select the value of the I/O register to edit in the [Value] area, then click on it again to switch the value to edit mode (press the [Esc] key to cancel the edit mode).
To write the edited value to the target memory, directly enter the value from the keyboard then press the [Enter] key.

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

Caution 2. The value of the read-only I/O register cannot be edited.

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. ASCII characters can be entered to the I/O register value.

- When the numeric "0x41" is written to the I/O register "OSTM*nXX*"
  >> "0x41" is written in the port "OSTM*nXX*".
- When the ASCII character "'A'" is written to the I/O register "OSTM*nXX*"
  >> "0x41" is written in the port "OSTM*nXX*".

2.12.3.5 Display/modify the I/O register contents during program execution

By registering an I/O register to the Watch panel as a watch-expression, the value of the I/O register can be displayed/modified not only while the program is stopped, but in execution.
See "2.12.6 Display/change watch-expressions" for details on the watch-expression.

2.12.3.6 Save the I/O register contents

The Save As dialog box can be opened by selecting the [File] menu >> [Save IOR Data As...], and all the contents of the I/O register can be saved in a text file (*.txt) or CSV file (*.csv). At this time, the values of all I/O registers become targets irrespective of the setting of display/non-display on this panel. When saving the contents to the file, the values of the I/O register 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.

Figure 2.50 Output Example of I/O register

```
IOR name          Value       Type (Byte Size)     Address
-------------------------------------------------------------
Category name
 -IOR name        Value       Type (Byte Size)     Address
    :              :             :                   :
```

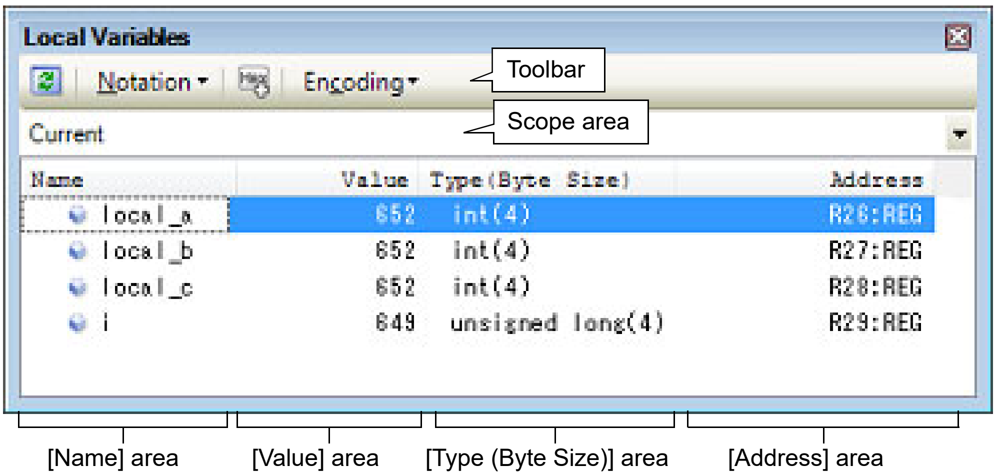
2.12.4 Display/change global variables/static variables

Global variables or static variables are displayed and its values can be changed in the Watch panel.
Register the variables to display/modify their values to the Watch panel as the watch-expressions.
For details, see "2.12.6 Display/change watch-expressions".

2.12.5 Display/change local variables

Contents of local variables can be displayed and its values can be changed in the Local Variables panel below.
Select the [View] menu >> [Local Variable].
Specify the scope in the scope area to display the contents of the target local variable.
In the Local Variables panel, the name of local variables and functions are displayed. The argument of the function is also displayed as the local variable.
For details on the contents and function in each area, see the section for the Local Variables panel.

Figure 2.51 Display the Contents of Local Variables (Local Variables Panel)



Caution Nothing is displayed on this panel during execution of a program.
When the program is stopped, items in each area are displayed.

This section describes the following.

2.12.5.1 Change display format of values
2.12.5.2 Modify the contents of local variables
2.12.5.3 Save the contents of local variables

2.12.5.1 Change display format of values

The display format of the [value] area can be changed using buttons below on the toolbar.

Notation	The following buttons to change the notation of a data value are displayed.
AutoSelect	Displays values on this panel in the default notation according to the type of variable (default).
Hexadecimal	Displays values on this panel in hexadecimal number.
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.
Decimal Notation for Array Index	Displays array indexes on this panel in decimal number (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel in hexadecimal number.
Float	Displays values on this panel in float. Note that when the value is not 4-byte data, or has the type information, displays it in the default notation.
Double	Displays values on this panel in double. Note that when the value is not 4-byte data, or has the type information, displays it in the default notation.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value.
Encoding	The following buttons to change the encoding of character variables are displayed.
ASCII	Displays character variables in ASCII code (default).
Shift_JIS	Displays character variables in Shift-JIS code.
EUC-JP	Displays character variables in EUC-JP code.
UTF-8	Displays character variables in UTF-8 code.
UTF-16 Big-Endian	Displays character variables in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character variables in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character variables in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character variables in UTF-32 Little-Endian code.

2.12.5.2 Modify the contents of local variables

Values and arguments of local variables can be edited.

Select the value of the local variables/arguments to edit in the [Value] area, then click on it again to switch the value to edit mode (press the [Esc] key to cancel the edit mode).

To write the edited value to the target memory, directly enter the value from the keyboard then press the [Enter] key. At this time, the edited value is checked and if it is incompatible with the type, the editing is invalidated.

Caution This operation cannot be performed during program execution.

Remark 1. If a number with fewer digits than the size of the variable is entered, the higher-order digits will be padded with zeroes.

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

Remark 3. If the display format of a character array (type char or unsigned char) is set to ASCII, then the value can also be entered as a string (ASCII/Shift-JIS/EUC-JP/Unicode (UTF-8/UTF-16 Big-Endian/UTF-16 Little-Endian/UTF-32 Big-Endian/UTF-32 Little-Endian)).

Remark 4. ASCII characters can be entered to values of local variables.

- Entering via an ASCII character
  In the [Value] area for the variable "ch", enter "'A'"
  >> "0x41" will be written to the memory area allocated to "ch"
- Entering via a numeric value
  In the [Value] area for the variable "ch", enter "0x41"
  >> "0x41" will be written to the memory area allocated to "ch"
- Entering via an ASCII string
  Set the display format of character array "str" to ASCII, and in the [Value] area, enter ""ABC""
  >> "0x41, 0x42, 0x43, 0x00" will be written to the memory area allocated to "str"

2.12.5.3 Save the contents of local variables

The Save As dialog box can be opened by selecting the [File] menu >> [Save Local Variables Data As...], and all the contents in the local variables can be saved in a text file (*.txt) or CSV file (*.csv).

When saving to files, retrieve the latest information from the debug tool. 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.

Figure 2.52 Output Example of Local Variables

```
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
         :                   :       :                  :
```

2.12.6 Display/change watch-expressions

By registering C language variables, CPU register, I/O register, and assembler symbols to the Watch panel as watch-expressions, you can always retrieve their values from the debug tool and monitor the values in batch.

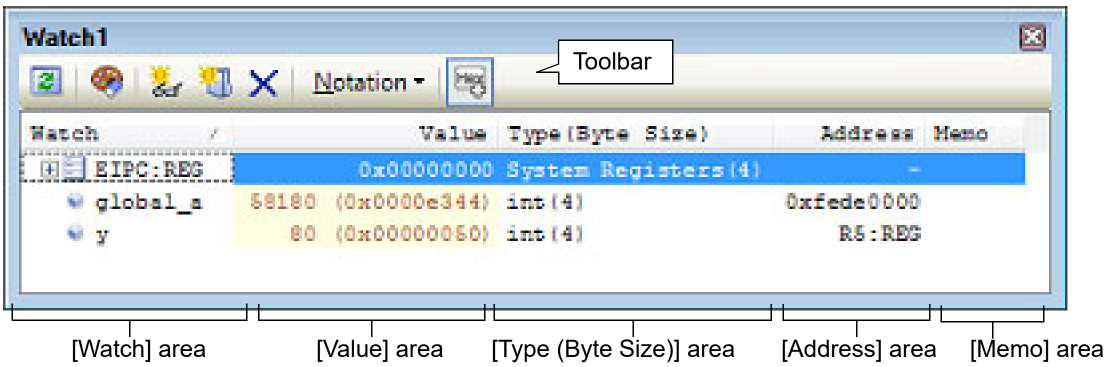
The values of watch-expressions can be updated during the program is in execution (see "2.12.6.7 Display/modify the contents of watch-expressions during program execution").

Select the [View] menu >> [Watch] >> [Watch*1 - 4*] to open the Watch panel.

The Watch panel can be opened up to 4 panels. Each panel is identified by the names "Watch1", "Watch2", "Watch3" and "Watch4" on the titlebar, and the watch-expressions can be registered/deleted/moved individually, and they are saved as the user information of the project.

For details on the contents and function in each area, see the section for the Watch panel.

Figure 2.53 Display the Contents of Watch-Expression (Watch Panel)



This section describes the following.

2.12.6.1 Register a watch-expression

There are three ways as follows to register watch-expressions (watch-expressions are not registered as default).

Caution 1. Watch-expressions can be registered up to 3000 in one watch panel (if this restriction is violated, a message appears).

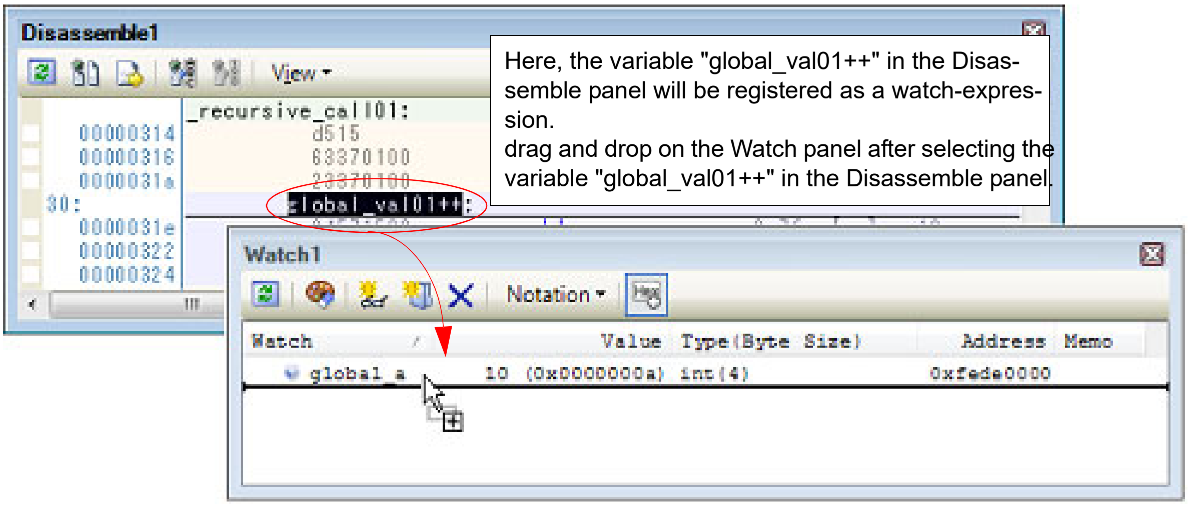
Caution 2. Due to compiler optimization, the data for the target variable may not be on the stack or in a register in blocks where that variable is not used. In such cases, if the variable is registered as a watch-expression, then the value will be displayed as a question mark "?".

Remark 1. Each watch-expression registered in each watch panel (Watch1 to Watch4) is managed in each panel and saved as the user information of the project.

Remark 2. More than one watch-expression with the same name can be registered.

(1) Register from other panels
Watch-expressions can be registered from other panel in CS+.
In other panel, drag and drop the watch-expression to register in any watch panel (Watch1 to Watch4).
For the relationship between panels that can use this operation and targets that can be registered as watch-expressions, see "Table A.2 Relationship between Panels and Targets That Can be Registered as Watch-Expressions".

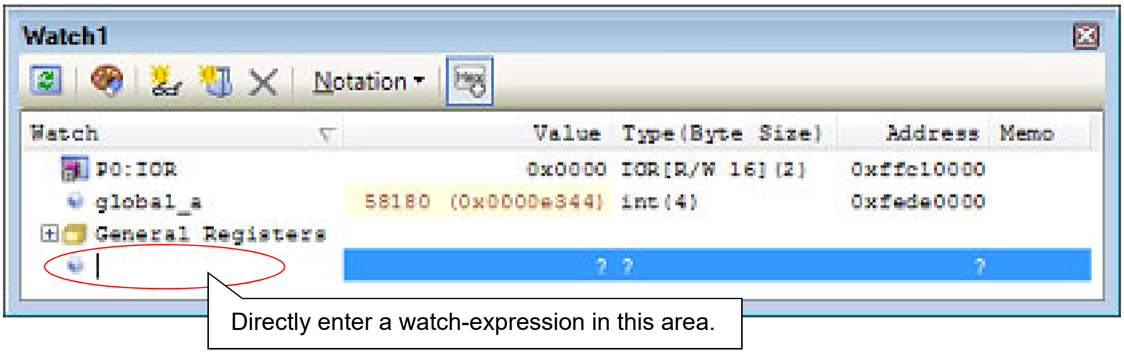
Figure 2.54 Registering Watch-Expressions from Other Panels



Remark You can also add a watch-expression by doing the following. First, select the target for which you wish to register a watch-expression, or move the caret to one of the target strings (the target is determined automatically). Next, from the context menu, select [Register Watch1] (but this is limited to the Watch panel (Watch 1)).

(2) Directly register in the Watch panel
Click the button in the toolbar in any the Watch panel (Watch1 to Watch4) to display the following entry box in the [Watch] area.

Figure 2.55 Entry Box of Watch-Expression



Directly input a watch-expression from the keyboard in the entry box then press the [Enter] key.
For the input forms of watch-expressions entered this way, see the tables listed below.

- "Table 2.37 Basic Input Format of Watch-expressions"
- "Table A.3 Handling of a C Language Function When Registered in Watch by Specifying Scope"
- "Table A.5 Handling of a CPU Register When Registered in Watch by Specifying Scope"
- "Table A.6 Handling of an I/O Register when Registered in Watch by Specifying Scope"

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this area (see "2.21.2 Symbol name completion function").

(3) Register from other application
Select the character strings of C language variables/CPU registers/I/O register/assembler symbols from an external editor and drag and drop it in the Watch panel (Watch 1 to Watch 4).
The dropped character strings are registered as a watch-expression.

2.12.6.2 Organize the registered watch-expressions

Registered watch-expressions can be organized in categories (folders) and displayed in tree view (there is no category as default).

Caution 1. Categories cannot be created within categories.

Caution 2. Up to 1500 categories can be created in one watch panel (if this restriction is violated, a message appears).

(1) Create new category
Move the caret to the position to create a new category then click the button in the toolbar and directly enter the new category name.

(2) Edit category name
Click the category name to edit, and click it again, then directly modify the category name from the keyboard.

(3) Delete categories
Select categories to delete then click the button in the toolbar.

(4) Change the display order
Registered watch-expressions are categorized when they are dragged and dropped in the category.
Also, the display order of the categories and the watch-expressions (upper or lower position) can be changed easily by drag and drop operation.

Remark Drag and drop the watch-expressions/categories in other watch panel (Watch1 to Watch4) to copy them.

2.12.6.3 Edit the registered watch-expressions

Registered watch-expressions can be edited.
Double-click the watch-expression to edit to switch the watch-expression to edit mode (press the [Esc] key to cancel the edit mode).

Directly edit from the keyboard and then press the [Enter] key.

2.12.6.4 Delete a watch-expression

To delete watch-expressions, select the one you want to delete in the Watch panel then click the button in the toolbar.

2.12.6.5 Change display format of values

The display format of the [value] area can be changed using buttons below on the toolbar.

Notation	The following buttons to change the notation of a data value are displayed.
AutoSelect	Displays the value of the selected watch-expression in the default notation (see "Table A.7 Display Format of Watch-Expressions (Default)") according the type of variable (default).
Hexadecimal	Displays the value of the selected item in hexadecimal number.
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned Decimal	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.
Float	Displays the value of the selected item in float. Note that this item becomes valid only when the selected watch-expression value is 4-byte data.
Double	Displays the value of the selected item in double. Note that this item becomes valid only when the selected watch-expression value is 8-byte data.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item (except the item displayed in hexadecimal number).

2.12.6.6 Modify the contents of watch-expressions

The values of watch-expressions can be edited.

Double-click the value of the watch-expression to edit in the [Value] area to switch the value to edit mode (press the [Esc] key to cancel the edit mode).

To write the edited value to the target memory, directly enter the value from the keyboard then press the [Enter] key.

Note that only those values that correspond one by one to variables of C language, CPU registers, I/O register or assembler symbols can be edited. In addition, read-only I/O register values cannot be edited.

This operation can be taken place while the program is in execution. See "2.12.1.4 Display/modify the memory contents during program execution" for details on how to operate it.

Remark 1. If a number with fewer digits than the size of the variable is entered, the higher-order digits will be padded with zeroes.

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

Remark 3. If the display format of a character array (type char or unsigned char) is set to ASCII, then the value can also be entered as a string (ASCII/Shift-JIS/EUC-JP/Unicode (UTF-8/UTF-16 Big-Endian/UTF-16 Little-Endian/UTF-32 Big-Endian/UTF-32 Little-Endian)).

Remark 4. ASCII characters can be entered to values of watch-expressions.

- Entering via an ASCII character
  In the [Value] area for the variable "ch", enter "'A'"
  >> "0x41" will be written to the memory area allocated to "ch"

- Entering via a numeric value
  In the [Value] area for the variable "ch", enter "0x41"
  >> "0x41" will be written to the memory area allocated to "ch"
- Entering via an ASCII string
  Set the display format of character array "str" to ASCII, and in the [Value] area, enter ""ABC""
  >> "0x41, 0x42, 0x43, 0x00" will be written to the memory area allocated to "str"

2.12.6.7 Display/modify the contents of watch-expressions during program execution

The Memory panel/Watch panel has the real-time display update function that can update/modify the display contents of the memory/watch-expression in real-time while executing the program.

Using the real-time display update function allows you to display/modify the value of memory/watch-expression not only while the program is stopped, but also in execution.

See "2.12.1.4 Display/modify the memory contents during program execution" for details on how to operate it.

2.12.6.8 Export/import watch-expressions

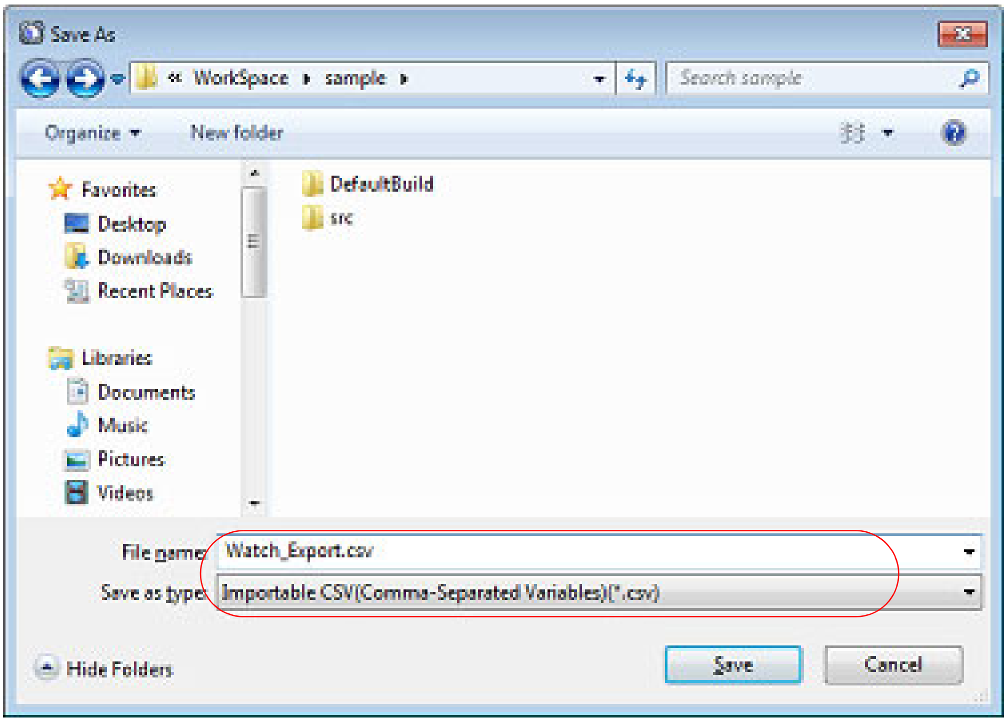
This feature is for the export of currently registered watch-expressions to a file and the importing of such files, enabling the re-registration of watch-expressions.

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.
With the Watch panel in focus, 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 2.56 Export of Watch-Expressions

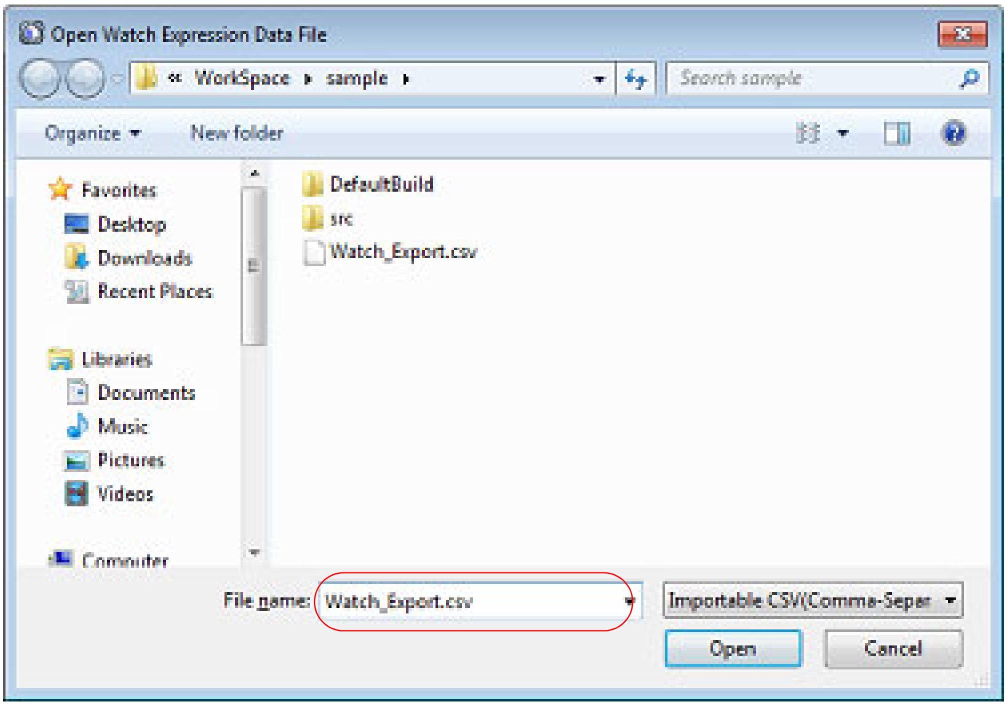


(2) Import watch-expressions
Import the file that exported in (1) to the Watch panel.

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 file that exported previously, 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 2.57 Import of Watch-Expressions



2.12.6.9 Save the 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 2.58 Output Example of Watch Data

```
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 When the contents of the panel are overwritten by selecting the [File] menu >> [Save Watch Data], each watch panel (Watch1 to Watch4) is treated as a different panel.

2.13 Display Information on Function Call from Stack

This section describes how to show the information on function call from the stack.

The CS+ compiler (CC-RH) pushes function-call information onto the stack, in accordance with the ANSI standard. It is thus possible to learn the function call depth, the location of the caller, parameters, and other information by analyzing the function-call information.

This "function-call information" is called the call stack information; this term will be used in the rest of this document.

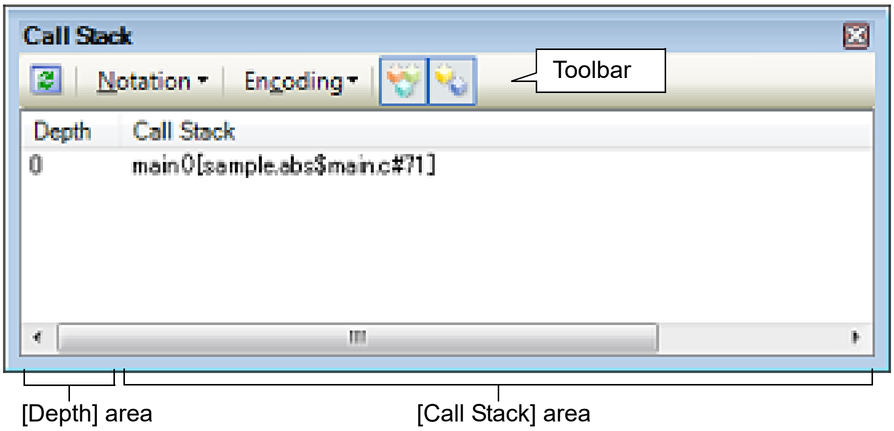
Remark For "call stack information" for a microcontroller that supports multi-core, see also to "Select a Core (PE)".

2.13.1 Display call stack information

Call stack information is displayed in the Call Stack panel below.
Select the [View] menu >> [Call Stack].
For details on the contents and function in each area, see the section for the Call Stack panel.

Caution Nothing is displayed on this panel during execution of a program.
When the program is stopped, items in each area are displayed.

Figure 2.59 Display Call Stack Information (Call Stack Panel)



This section describes the following.

2.13.1.1 Change display format of values

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

Notation	The following buttons to change the notation of a data value are displayed.
AutoSelect	Displays values on this panel in the default notation according to the type of variable (default).
Hexadecimal	Displays values on this panel in hexadecimal number.
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.

Encoding	The following buttons to change the encoding of character variables are displayed.
ASCII	Displays character variables in ASCII code (default).
Shift_JIS	Displays character variables in Shift-JIS code.
EUC-JP	Displays character variables in EUC-JP code.
UTF-8	Displays character variables in UTF-8 code.
UTF-16 Big-Endian	Displays character variables in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character variables in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character variables in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character variables in UTF-32 Little-Endian code.

2.13.1.2 Jump to the source line

Double-clicking on the line will open the Editor panel with the caret moved to the source line of the calling function indicated by the selected line (If the panel is already open, the screen will jump to the editor panel).

Remark Selecting [Jump to Disassemble] from the context menu will open the Disassemble panel (Disassemble 1) with the caret moved to address of the calling function indicated by the selected line (If the panel is already open, the screen will jump to the Disassemble panel (Disassemble 1)).

2.13.1.3 Display local variables

Selecting [Jump to Local Variable at This Time] from the context menu will open the Local Variables panel that displays the local variables indicated by the currently selected line.

2.13.1.4 Save the contents of call stack information

By selecting the [File] menu >> [Save Call Stack Data As...], the Save As dialog box can be opened, and all the contents in the call stack information can be saved in a text file (*.txt) or CSV file (*.csv).

When saving to files, retrieve the latest information from the debug tool.

Figure 2.60 Output Example of Call Stack Information

```
Depth      Call stack
----------------------------------------------------------------------
 0         Call stack information
 1         Call stack information
 :                :
```

2.14 Collect Execution History of Programs

This section describes how to collect the execution history of the program.
A history of program execution is generally called a trace; this term will be used in the remainder of this document.
It is nearly impossible to find the cause of runaway program execution from the memory contents, stack information, and the like after the runaway has occurred. The collected trace data, however, can be used to trace program execution up to the runaway directly, making this an effective tool for discovering hidden bugs.

Remark For "collection of execution history of programs" for a microcontroller that supports multi-core, see also to "2.9 Select a Core (PE)".

2.14.1 Configure the trace operation

When the trace function starts, 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 depending on the debug tool used.

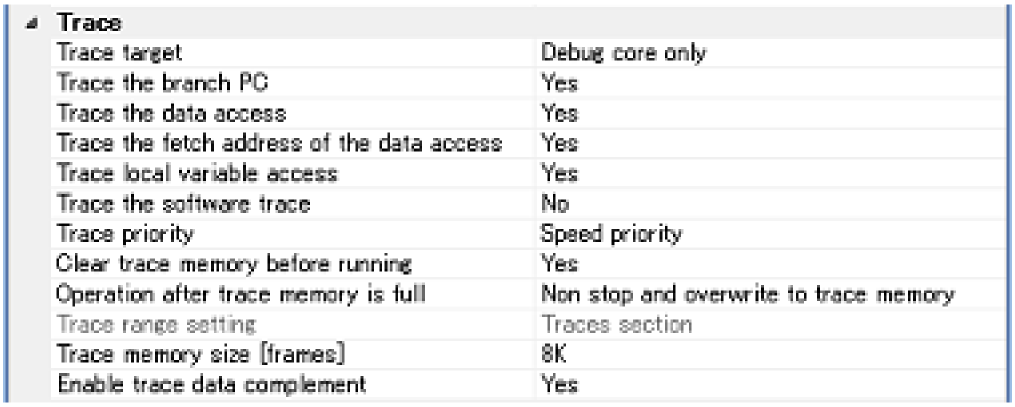
2.14.1.1 [Full-spec emulator]
2.14.1.2 [IE850A]/[E1]/[E20]/[E2]
2.14.1.3 [Simulator]

2.14.1.1 [Full-spec emulator]

This trace operation can be configured in the [Trace] category on the [Debug Tool Settings] tab in the Property panel.

Caution Properties in this category cannot be changed during program execution.

Figure 2.61 [Trace] Category [Full-spec emulator]



(1) [Trace target]
Select the trace target from the following drop-down list.
The choices of this property vary depending on the mode selected by the [Debug mode] property in the [Multi-core] category on the [Debug Tool Settings] tab of the Property panel.

- When Sync debug mode is selected:

Debug core only	Collects the trace data regarding the currently selected PE (default). After collecting the trace data, the contents of the Trace panel is not changed even if you switch PE.
All core	Collects the trace data for all PEs. After collecting the trace data, if you switch PE, the contents of the Trace panel will be updated to the corresponding trace data.

- When Async debug mode is selected:

core name	Collects the trace data for the selected *core name*.

This property appears only when the selected microcontroller is a multi-core.

This property can be changed only while all cores are stopped.

Caution This property can be changed only while all cores are stopped.
This property setting takes effect before user program execution is started.

(2) Select trace data
Select the type of trace data to be collected from the following properties.

Trace the branch PC	PC values for source/destination instructions of branching during program execution are collected as trace data.
Trace the data access	Data information on access-related events that occurred during program execution are collected as trace data.
Trace the fetch address of the data access	PC values for instructions of access-related events that occurred during program execution are collected as trace data.
Trace local variable access	Data information on access-related events for accesses to local variables that occurred during program execution is collected as trace data.
Trace the software trace	Information on trace output instructions to be embedded that were generated during program execution is collected as trace data.
Trace the DBCP	Information on DBCP that were generated during program execution is collected as trace data.
Trace the DBTAG	Information on DBTAG that were generated during program execution is collected as trace data.
Trace the fetch address of the DBTAG	Information on DBTAG that were generated during program execution is collected, along with the values of addresses where the DBTAG instructions were executed.
Trace the DBPUSH	Information on DBPUSH that were generated during program execution is collected as trace data.
Trace the fetch address of the DBPUSH	Information on DBPUSH that were generated during program execution is collected, along with the values of addresses where the DBTAG instructions were executed.

Caution The trace memory is cleared when you change the settings of these properties.

(3) [Trace priority]
Select which item should be given priority when using the trace function from the following drop-down list.

Speed priority	Traces giving priority to the real-time performance (default).
Data priority	Traces after stopping the execution pipeline of the CPU temporarily so that no data is missed.

Caution The trace memory is cleared when you change the setting of this property.

(4) [Clear trace memory before running]
Select whether to clear (initialize) the trace memory before tracing starts.
Select [Yes] to clear the memory (default).

Remark You can forcibly clear the trace memory when clicking the  button in the toolbar in the Trace panel.

(5) [Operation after trace memory is full]
Select the operation after the trace memory is full with the collected trace data from the following drop-down list.

Non stop and overwrite to trace memory	Continues overwriting the older trace data after the trace memory is full (default). When the [Clear trace memory before running] property is set to [Yes], at the time of a resumption, trace data is collected after clearing the trace memory.
Stop trace	When the trace memory is full, CS+ stops writing trace data (the program does not stop execution).
Stop	When the trace memory is full, CS+ stops writing trace data and the program stops execution.

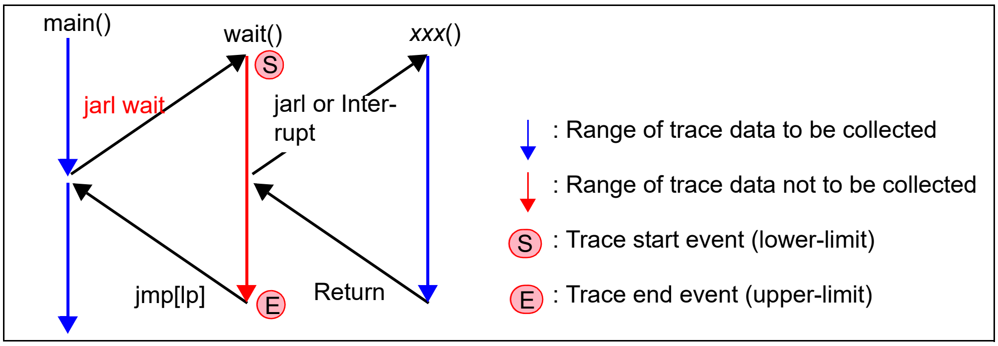
Caution The trace memory is cleared when you change the setting of this property.

(6) [Trace range setting]
Select the range of trace data to be collected from the following drop-down list.
Note, however, that this property can be changed only when connected to the debug tool.

Traces section	Collects the execution history as trace data within the section specified with a trace start event and a trace end event (default).
Traces out of range	Collects the execution history as trace data outside the range specified with a trace start event and a trace end event.

Caution If this property is changed, then all trace start and trace end events currently being set will become invalid.

Remark When [Traces out of range] is selected, the range of trace data to be collected will be determined by a lower-limit address and an upper-limit address that are specified with a trace start event and a trace end event.



Caution 1. For an out-of-range trace event, be sure to specify the start and end addresses.

Caution 2. Only one section is specifiable for an out-of-range trace event.

(7) [Trace memory size[frames]]
Specify from the drop-down list the size of trace memory (i.e. the number of trace frames) in this property.
The trace frame is a unit of trace data. One trace frame is used for each operation in fetch/write/read (default: [8K]).

Caution The trace memory is cleared when you change the setting of this property.

(8) [Enable trace data complement]
Select whether to enable complement display when displaying the collected trace data in the Trace panel.
By enabling complement display, instructions between branch instructions that cannot be traced by hardware can be displayed.
Select [Yes] to enable complement display (default).
This setting will be applied from the next acquisition of trace data.

2.14.1.2 [IE850A]/[E1]/[E20]/[E2]

For the IE850A, the external trace configuration is set up in the [Trace] category on the [Connect Settings] tab in the Property panel.

Caution 1. If the settings for external tracing are not suitable for the microcontroller or the emulator, external tracing will not be available.

Caution 2. All properties in this category become unchangeable after connecting to the debug tool.

Figure 2.62 [Trace] Category [IE850A]

Trace	
Use external trace	Yes
Number of lanes of external trace	4
Transfer speed of external trace [Gbps]	Auto

(1) [Use external trace]
Select whether to use the external trace.

(2) [Number of lanes of external trace]
Select the number of lanes of the external trace.

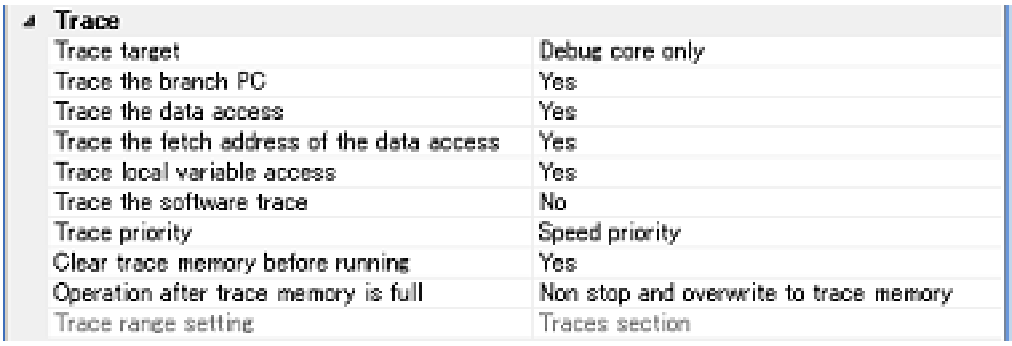
(3) [Transfer speed of external trace [Gbps]]
Select the transfer rate for the external trace.
When [Auto] is selected, this will be the fastest value for the selected microcontroller.

This trace operation can be configured in the [Trace] category on the [Debug Tool Settings] tab of the Property panel.

Caution 1. If the trace function is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (the trace function cannot be used).

Caution 2. Properties in this category cannot be changed during program execution.

Figure 2.63 [Trace] Category [E1][E20]



(1) [Trace target]
Select the trace target from the following drop-down list.
The choices of this property vary depending on the mode selected by the [Debug mode] property in the [Multi-core] category on the [Debug Tool Settings] tab of the Property panel.

- When Sync debug mode is selected:

Debug core only	Collects the trace data regarding the currently selected PE (default). After collecting the trace data, the contents of the Trace panel is not changed even if you switch PE.
All core	Collects the trace data for all PEs. After collecting the trace data, if you switch PE, the contents of the Trace panel will be updated to the corresponding trace data.

- When Async debug mode is selected:

core name	Collects the trace data for the selected *core name*.

This property appears only when the selected microcontroller is a multi-core.
This property can be changed only while all cores are stopped.

Caution This property can be changed only while all cores are stopped.
This property setting takes effect before user program execution is started.

(2) Set the system protection ID (SPID) filter
When the SPID filter is in use, only information on tracing which belongs to the SPIDs specified with the [SPID filter] property is collected. Set the SPID filter in the Select SPID filter dialog box.

(3) Select trace data
Select the type of trace data to be collected from the following properties.

Trace the branch PC	PC values for source/destination instructions of branching during program execution are collected as trace data.
Trace the data access	Data information on access-related events that occurred during program execution are collected as trace data.

Trace the fetch address of the data access	PC values for instructions of access-related events that occurred during program execution are collected as trace data.
Trace local variable access	Data information on access-related events for accesses to local variables that occurred during program execution is collected as trace data.
Trace the transition information of CPU operation mode	Transitions of the CPU operating mode that occur during the execution of a program are recorded as trace data. If this is selected, the CPU operating mode is indicated for all trace data.
Trace the software trace	Information on trace output instructions to be embedded that were generated during program execution is collected as trace data.
Trace the DBCP	Information on DBCP that were generated during program execution is collected as trace data.
Trace the DBTAG	Information on DBTAG that were generated during program execution is collected as trace data.
Trace the fetch address of the DBTAG	Information on DBTAG that were generated during program execution is collected, along with the values of addresses where the DBTAG instructions were executed.
Trace the DBPUSH	Information on DBPUSH that were generated during program execution is collected as trace data.
Trace the fetch address of the DBPUSH	Information on DBPUSH that were generated during program execution is collected, along with the values of addresses where the DBTAG instructions were executed.
Trace the GTM	The results of tracing of the GTM are collected.
Trace the branch PC of the MCS	PC values for the source and destination instructions in branch processing that occurred during GTM execution are collected as trace data.
Trace the data access of the MCS	Data information on access-related events that occurred during GTM execution is collected as trace data.

Caution The trace memory is cleared when you change the setting of these properties.

(4) [Trace priority]
Select which item should be given priority when using the trace function from the following drop-down list.

Speed priority	Traces giving priority to the real-time performance (default).
Data priority	Traces after stopping the execution pipeline of the CPU temporarily so that no data is missed.

Caution 1. The trace memory is cleared when you change the setting of this property.

Caution 2. When [Data priority] is selected, the function to stop tracing ([Stop trace] in the [Operation after trace memory is full] property) is not usable.

(5) [Clear trace memory before running]
Select whether to clear (initialize) the trace memory before tracing starts.
Select [Yes] to clear the memory (default).

Remark You can forcibly clear the trace memory when clicking the  button in the toolbar in the Trace panel.

(6) [Operation after trace memory is full]
Select the operation after the trace memory is full with the collected trace data from the following drop-down list.

Non stop and overwrite to trace memory	Continues overwriting the older trace data after the trace memory is full (default). When the [Clear trace memory before running] property is set to [Yes], at the time of a resumption, trace data is collected after clearing the trace memory.
Stop trace	When the trace memory is full, CS+ stops writing trace data (the program does not stop execution).

Stop	When the trace memory is full, CS+ stops writing trace data and the program stops execution.

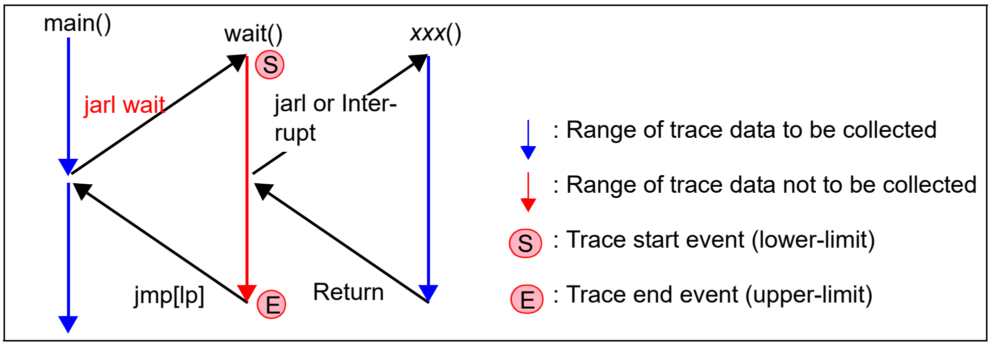
Caution The trace memory is cleared when you change the setting of this property.

(7) [Trace range setting]
Select the range of trace data to be collected from the following drop-down list.
Note, however, that this property can be changed only when connected to the debug tool.

Traces section	Collects the execution history as trace data within the section specified with a trace start event and a trace end event (default).
Traces out of range	Collects the execution history as trace data outside the range specified with a trace start event and a trace end event.

Caution If this property is changed, then all trace start and trace end events currently being set will become invalid.

Remark When [Traces out of range] is selected, the range of trace data to be collected will be determined by a lower-limit address and an upper-limit address that are specified with a trace start event and a trace end event.



Caution 1. For an out-of-range trace event, be sure to specify the start and end addresses.

Caution 2. Only one section is specifiable for an out-of-range trace event.

(8) [Trace memory size[frames]] [IE850A]
Specify from the drop-down list the size of trace memory (i.e. the number of trace frames) in this property.
The trace frame is a unit of trace data. One trace frame is used for each operation in fetch/write/read (default: [8K]).

Caution The trace memory is cleared when you change the setting of this property.

Software tracing through LPD output can be configured in the [Trace] category on the [Debug Tool Settings] tab of the Property panel.

Caution 1. If software tracing through LPD output is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (software tracing through LPD output cannot be used).

Caution 2. Properties in this category cannot be changed during program execution.

Figure 2.64 [Output Software Trace from LPD] Category [E2]

Output Software Trace from LPD	
Output the software trace from the LPD	**Yes**
Target when outputting the software trace from the LPD	CPU1
Output the DBCP	Yes
Output the DBTAG	Yes
Output the fetch address of the DBTAG	Yes
Output the DBPUSH	Yes
Output the fetch address of the DBPUSH	Yes
Priority when outputting the software trace data from the LPD	Speed priority
Operation after the recording memory is full	Overwrite the record memory and continue

(1) [Output the software trace from the LPD]
Select whether to output the software trace from the LPD.
When [No] is selected, you cannot change this property while CS+ is connected to E2.

(2) [Target when outputting the software trace from the LPD]
Select the target when outputting the software trace from the LPD from the following drop-down list.
This property appears only when the selected microcontroller is a multi-core.
You cannot change this property while CS+ is connected to E2.

core name	Outputs the software trace data from the LPD for the selected *core name*.

(3) Select the software trace data
Select the type of the software trace data to be output from the LPD from the following properties.

Output the DBCP	Information on DBCP that were generated during program execution is output as trace data from the LPD.
Output the DBTAG	Information on DBTAG that were generated during program execution is output as trace data from the LPD.
Output the fetch address of the DBTAG	Information on DBTAG that were generated during program execution is output from the LPD, along with the values of addresses where the DBTAG instructions were executed.
Output the DBPUSH	Information on DBPUSH that were generated during program execution is output as trace data from the LPD.
Output the fetch address of the DBPUSH	Information on DBPUSH that were generated during program execution is output from the LPD, along with the values of addresses where the DBTAG instructions were executed.

(4) [Priority when outputting the software trace data from the LPD]
Select which item should be given priority when using software tracing through LPD output from the following drop-down list.

Speed priority	Outputs the software trace from the LPD giving priority to the real-time performance (default).
Data priority	Outputs the software trace from the LPD after stopping the execution pipeline of the CPU temporarily so that no data is missed.

(5) [Operation after the recording memory is full]
Select the operation after the recording memory is full with software trace data from the following drop-down list.

Overwrite the record memory and continue	Continues overwriting older software trace data even after the recording memory is used up (default).
Stop recording	Stops outputting software trace data when the recording memory is used up (the program execution will not be stopped).
Stop program	Stops running the program and outputting software trace data when the recording memory is used up.

2.14.1.3 [Simulator]

This trace operation can be configured in the [Trace] category on the [Debug Tool Settings] tab of the Property panel.

Figure 2.65 [Trace] Category [Simulator]

Trace	
Use trace function	No
Trace the branch PC and the data access	Yes
Trace the software trace	**Yes**
Trace the DBCP	Yes
Trace the DBTAG	Yes
Trace the DBPUSH	Yes
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]	4K
Rate of frequency division of trace time tag	1/1

(1) [Use trace function]
Select whether to use trace function.
Select [Yes] to use the trace function (default: [No]).

(2) [Trace target]
Select the trace target from the following drop-down list.

Debug core only	Collects the trace data regarding the currently selected PE (default). After collecting the trace data, if you switch PE, the contents of the Trace panel will be updated to the corresponding trace data.
All core	Collects the trace data for all PEs. After collecting the trace data, the contents of the Trace panel is not changed even if you switch PE.

(3) Select trace data
Select the type of trace data to be collected from the following properties.

Trace the branch PC and the data access	PC values for the source and destination instructions of branching during program execution and the PC values and information on the data for instructions leading to access-related events that occur during program execution are collected as trace data.
Trace the software trace	Information on trace output instructions to be embedded that were generated during program execution is collected as trace data.
Trace the DBCP	Information on DBCP that were generated during program execution is collected as trace data.
Trace the DBTAG	Information on DBTAG that were generated during program execution is collected as trace data.
Trace the DBPUSH	Information on DBPUSH that were generated during program execution is collected as trace data.

Caution The trace memory is cleared when you change the setting of these properties.

(4) [Clear trace memory before running]
Select whether to clear (initialize) the trace memory before tracing starts.
Select [Yes] to clear the memory (default).

Remark You can forcibly clear the trace memory when clicking the button in the toolbar in the Trace panel.

(5) [Operation after trace memory is full]
Select the operation after the trace memory is full with the collected trace data from the following drop-down list.

Non stop and overwrite to trace memory	Continues overwriting the older trace data after the trace memory is full (default). When the [Clear trace memory before running] property is set to [Yes], at the time of a resumption, the trace data is collected after clearing the trace memory.

Stop trace	When the trace memory is full, CS+ stops writing trace data (the program does not stop execution).
Stop	When the trace memory is full, CS+ stops writing trace data and the program stops execution.

(6) [Accumulate trace time]
Select whether to display the trace time with accumulated time.
Select [Yes] to display trace time with accumulated time. Select [No] to display the trace time with differential time (default).

(7) [Trace memory size[frames]]
Select the trace memory size (trace frame number).
The trace frame is a unit of trace data. One trace frame is used for each operation in fetch/write/read.
Drop down list includes the following trace frame numbers.
4K (default), 8K, 12K, 16K, 20K, 24K, 28K, 32K, 36K, 40K, 44K, 48K, 52K, 56K, 60K, 64K, 128K 192K, 256K, 320K, 384K, 448K, 512K, 576K, 640K, 704K, 768K, 832K, 896K, 960K, 1M, 2M, 3M

(8) [Rate of frequency division of trace time tag]
Select the frequency division ratio of the counter to be used for time tag display (the [Time] item in the Trace panel) (default: [1/1]).
When the division ratio is set, the number of clocks required for the incrementation of the counter displayed on the time tag of the branch PC and the data access is changed.

2.14.2 Collect execution history until stop of the execution

In the debug tool, there is a function to collect the execution history from the start of program execution to the stop.
Therefore, the trace data collection is automatically started when the program starts executing and stopped when the program stops.
See "2.14.6 Display the collected execution history" for how to check the collected trace data.

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.
Consequently, if the Unconditional Trace event is set to Invalid state by clearing the check box in the Events panel, trace data linked to the start of program execution will not be collected (the Unconditional Trace event is set to Valid state by default).
Note that Unconditional Trace event and Trace event described later (see "2.14.3 Collect execution history in a section") are used exclusively of each other. Therefore, if Trace event with Valid state is set, Unconditional Trace event is automatically set to Invalid state.

2.14.3 Collect execution history in a section

The execution history is collected as trace data only for a section while the program is in execution by setting a Trace event.
This Trace event consists of a trace start event and a trace end event.
To use this function, follow the procedure described below.

Caution 1. Also see "2.19.7 Notes for setting events" for details on Trace events (e.g. limits on the number of enabled events).

Caution 2. [Simulator]
Trace start events and trace stop events cannot be set/deleted while a tracer is running.

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.

2.14.3.1 Set a Trace event

To set a Trace event, set a trace start event and a trace end event that starts/stops collecting the trace data.
Use one of the following methods to set a trace start event and a trace end event.

(1) For execution-related events
By setting execution-related events for a trace start event and a trace end event, it is possible to start and stop the collection of trace data at any place.
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, in accordance with your desired event type, after moving the caret to the target line that has a valid address.

Event Type	Operation
Trace start	Select [Trace Settings] >> [Start Tracing]
Trace end	Select [Trace Settings] >> [Stop Tracing]

Caution [Simulator]
Simulator will not display a trace end event as the results of a trace. For this reason, set a trace end event to one line below the range that you wish to display as the trace data.

A trace start event or a trace end event is set to the instruction at the start address corresponding to the line of the caret position. Once a trace start event or a trace end event is set, the following event mark is displayed in the event area of the line/address that an event is set.

Table 2.14 Event Marks of Trace Start Event and Trace End Events

Event Type	Event Mark
Trace start	
Trace end	

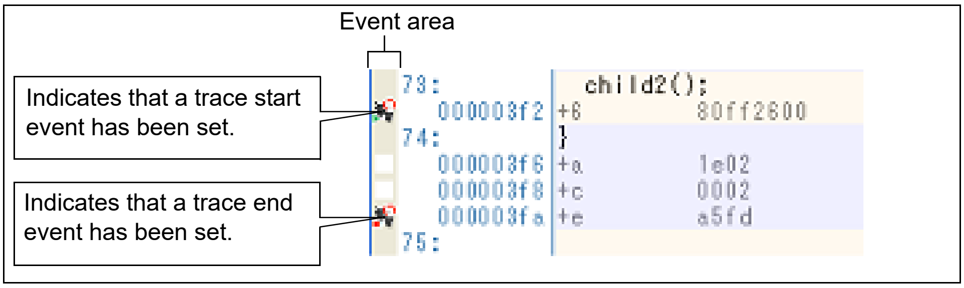


Figure 2.66 Trace Start and Trace End Events Setting Example (Disassemble Panel)

Remark [Full-spec emulator][E1][E20]
By specifying [Traces out of range] in the [Trace range setting] property in the [Trace] category on the [Debug Tool Settings] tab of the Property panel, you can collect the execution history as trace data outside the specified range.

(2) For access-related events [Full-spec emulator][E1][E20]
By setting access-related events for a trace start event and a trace end event, it is possible to start and stop the collection of trace data when a specified access is made to any variable or IOR.

Caution The types of access that can be set by using methods described here are only a read/write (see "Table 2.8 Types of Accesses to Variables").

(a) To set events for variables or IOR in the Editor panel/Disassemble panel
Perform this operation in the Editor panel/Disassemble panel in which the source text/disassembly text is displayed.
Follow the operation listed below from the context menu, in accordance with your desired event type, after selecting an arbitrary variable or IOR on the source text/disassembled text.
Note, however, that only global variables, static variables inside functions, and file-internal static variables can be used.

Event Type	Operation
Trace start	Select [Trace Settings] >> [Record Start R/W Value], and then press the [Enter] key.
Trace end	Select [Trace Settings] >> [Record End R/W Value], and then press the [Enter] key.

At this time, if you have specified a value in the text box in the context menu, collection of trace data is started or finished only when a read/write is performed with a specified value. If no value is specified, collection of trace data is started or finished when a read/write is performed to the selected variable or IOR, regardless of the value.

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

Caution 2. Variables or SFR at lines that have no valid addresses cannot be used for trace start events and trace end events.

(b) To set events for registered watch-expressions
Perform this operation in the Watch panel.
Follow the operation listed below from the context menu after selecting the registered watch-expression (multiple selections not allowed). Note, however, that only global variables, static variables inside functions, file-internal static variables, and IOR can be used.

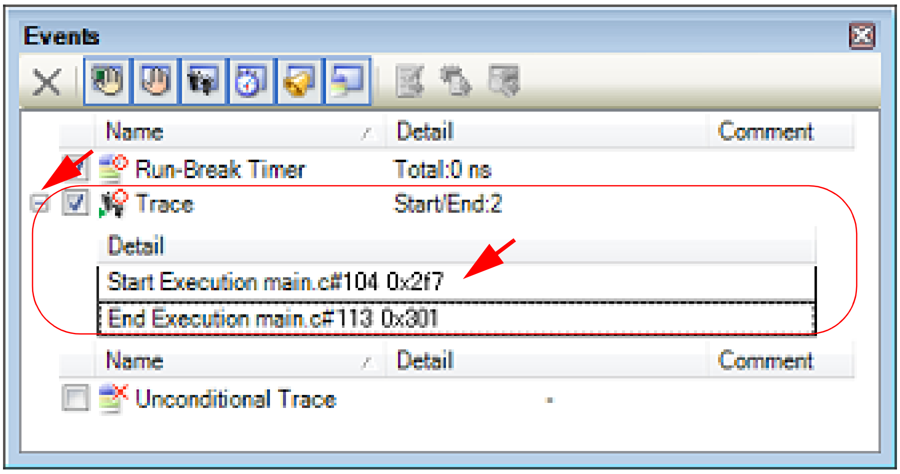
Event Type	Operation
Trace start	Select [Trace Output] >> [Record Start R/W Value], and then press the [Enter] key.
Trace end	Select [Trace Output] >> [Record End R/W Value], and then press the [Enter] key.

At this time, if you have specified a value in the text box in the context menu, collection of trace data is started or finished only when a read/write is performed with a specified watch-expression. If no value is specified, collection of trace data is started or finished when a read/write is performed to the selected variable or IOR, regardless of the value.

Caution 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.

When a trace start event and a trace end event are set, they are managed collectively on the Events panel as one instance of a Trace event (see "2.19 Manage Events"). When you click the "+" mark at a Trace event item, detailed information on the trace start event and the trace end event you have set is displayed.

Figure 2.67 Example of Trace Start and Trace End Events (Execution Type) in Events Panel



Remark 1. If either one of a trace start event and a trace end event is set as Valid state, the check box of Unconditional Trace event in the Events panel is automatically cleared, therefore, trace data collection does not automatically start with the start of the program execution (the tracer will not run until the condition of the trace start event that has been set is met).

Remark 2. A trace end event is not indispensable for a Trace event.

Remark 3. Event marks differ depending on the event state (see "2.19.1 Change the state of set events (valid/invalid)").
When an event is set at the point which other event is already set, the event mark ( ) is displayed meaning more than one event is set at the point.

Remark 4. [Simulator]
If either one of a trace start event and a trace end event is set to Valid state, the [Use trace function] property in the [Trace] category on the [Debug Tool Settings] tab of the Property panel is automatically set to [Yes] and the trace function will be enabled.

2.14.3.2 Execute the program

Execute the program (see "2.10 Execute Programs").
Collection of trace data is started or finished when the condition set for a trace start event or a trace end event is met.
See "2.14.6 Display the collected execution history" for how to check the collected trace data.

2.14.3.3 Edit a Trace event

It is possible to edit a trace start event or a trace end event you have set.
For details on how to do it, see "2.19.4.1 Edit execution-related events" or “2.19.4.2 Edit access-related events”.

2.14.3.4 Delete a Trace event

To delete a Trace event you have set, on the Editor panel/Disassemble panel, right-click the event mark in the event area and select [Delete Event] from the context menu that is displayed.
Also, there is another way to delete a set event. Select the Trace event you want to delete on the Events panel, and then click the button in the toolbar (see "2.19.5 Delete events").

Caution It is not possible to delete only a trace start event or a trace end event (i.e. if either a trace start event or a trace end event is deleted from the event marks on the event area, all of the corresponding event marks are deleted).

2.14.4 Collect execution history only when the condition is met

The program execution history can be collected only when a condition is met.
By setting a Point Trace event, the execution history is collected as trace data only when an arbitrary variable or I/O register is accessed with the specified type.
To use this function, follow the procedure described below.

2.14.4.1 Set a Point Trace event
2.14.4.2 Execute the program
2.14.4.3 Edit a Point Trace event
2.14.4.4 Delete a Point Trace event

2.14.4.1 Set a Point Trace event

Use one of the following methods to set a Point Trace event.

Caution 1. Also see "2.19.7 Notes for setting events" for details on Point Trace events (e.g. limits on the number of enabled events).

Caution 2. You cannot set/delete Point Trace events while a tracer is running.

Caution 3. Accesses via DMA cannot be traced.

Caution 4. [Full-spec emulator][E1][E20]
When a Point Trace event is used simultaneously with a trace event (trace start event or trace end event) that collects the execution history of a section, the execution history of Point Trace is not collected until the condition of the trace start event is met.

Remark [Simulator]
When a Point Trace event is set to Valid state, the [Use trace function] property in the [Trace] category on

the [Debug Tool Settings] tab of the Property panel is automatically set to [Yes] and the trace function will be enabled.

(1) For access to a variable or I/O register on the Editor panel/Disassemble panel
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, in accordance with your desired access type, after selecting the variable or I/O register as the subject to access.
Note, however, that only global variables, static variables inside functions, and file-internal static variables can be used.

Access Type	Operation
Read	Select [Trace Settings] >> [Record Reading Value].
Write	Select [Trace Settings] >> [Record Writing Value].
Read/Write	Select [Trace Settings] >> [Record R/W Value].

Caution Variables within the current scope can be specified.

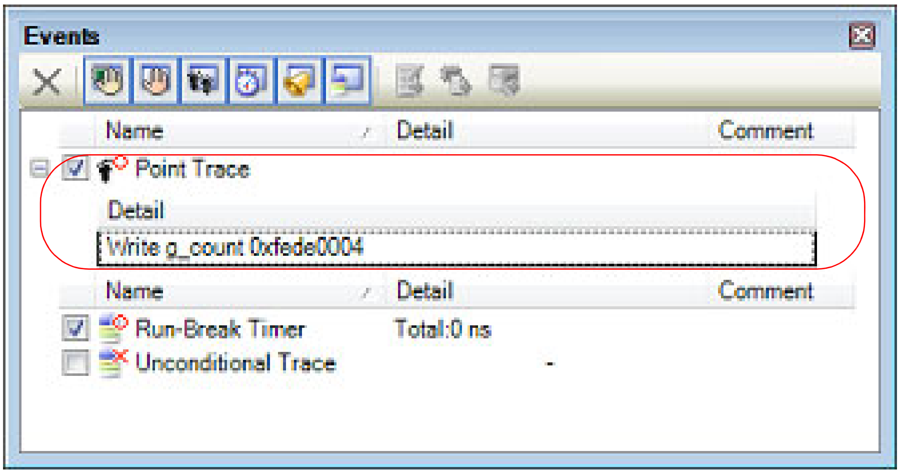
(2) For access to a registered watch-expression
Perform this operation in the Watch panel.
Select the watch-expression as the subject to access and perform the following operation from the context menu (see "2.12.6 Display/change watch-expressions").
Note, however, that only global variables, static variables inside functions, file-internal static variables, and I/O register can be used.

Access Type	Operation
Read	Select [Trace Output] >> [Record Reading Value].
Write	Select [Trace Output] >> [Record Writing Value].
Read/Write	Select [Trace Output] >> [Record R/W Value].

Caution 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.

By performing the above operation, it is interpreted as if a Point Trace event has been set at the target variable/I/O register/watch-expression, and it is managed in the Events panel (see "2.19 Manage Events" for details).

Figure 2.68 Example of Setting Point Trace Event in Events Panel



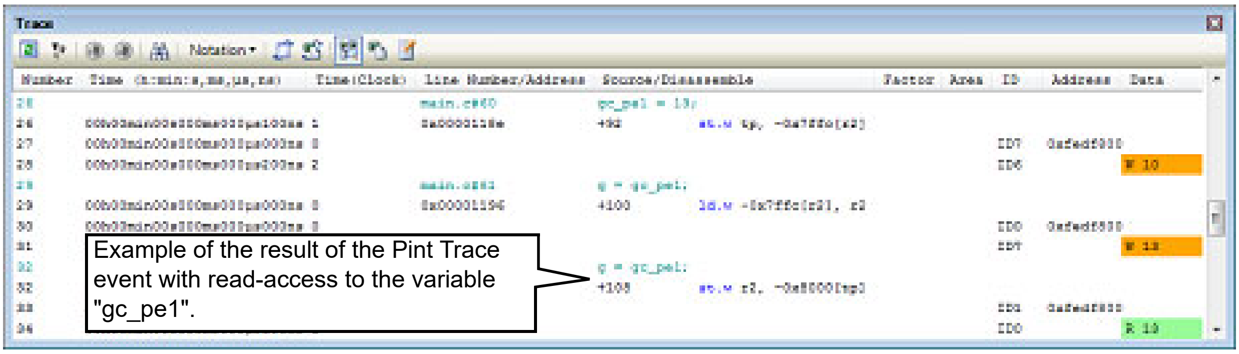
2.14.4.2 Execute the program

Execute the program (see "2.10 Execute Programs").

If the conditions for a Point Trace event that you have set are met while the program is executing, that information is collected as trace data.

See "2.14.6 Display the collected execution history" for details on checking trace data.

Figure 2.69 Example of Point Trace Event Results View



2.14.4.3 Edit a Point Trace event

Detailed information on a Point Trace event is edited in the Detailed Settings of Point Trace dialog box. This dialog box is opened by selecting the Point Trace event you desire to edit on the Events panel then selecting [Edit Condition...] from the context menu.

(1) When editing an address condition
The compare condition, address and address mask can be specified as an address condition in the Detailed Settings of Point Trace dialog box.

(2) When editing an data condition
The access type, access size, compare condition, compare data and data mask can be specified as an data condition in the Detailed Settings of Point Trace dialog box.

(3) When editing an pass count condition [Simulator]
The pass count can be specified as an pass count condition in the Detailed Settings of Point Trace dialog box.

(4) When editing an event target [Full-spec emulator][E1][E20]
The following can be specified as an event target in the Detailed Settings of Point Trace dialog box.

Event Target	Function
CPU	Access to the CPU is the target of the point trace (default).
Global RAM	Access to the Global RAM is the target of the point trace.
Cluster RAM	Access to the Cluster RAM is the target of the point trace.

2.14.4.4 Delete a Point Trace event

To delete a Point Trace event you have set, select the Point Trace event you want to delete on the Events panel, and then click the button in the toolbar (see "2.19.5 Delete events").

2.14.5 Stop/restart collection of execution history

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

2.14.5.1 Stop collection of execution history temporarily
2.14.5.2 Restart collection of execution history

2.14.5.1 Stop collection of execution history temporarily

By clicking the button on the toolbar in the Trace panel during program execution, it is possible to temporarily stop collection of trace data without stopping program execution.

Use this function 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.

2.14.5.2 Restart collection of execution history

If you have halted the trace function during program execution, you can start collection of trace data again by clicking the button on the toolbar in the Trace panel.
Note that the trace data that has been collected before you restart is cleared once.

2.14.6 Display the collected execution history

The collected trace data is displayed in the Trace panel below.
Select the [View] menu >> [Trace].
The trace data displays by mixing the disassembled text and source text by default, but it is also possible to display either one of these by selecting the Display mode.
For details on the contents and function in each area, see the section for the Trace panel.

Figure 2.70 Display Trace Data (Trace Panel [Full-spec emulator][E1][E20])

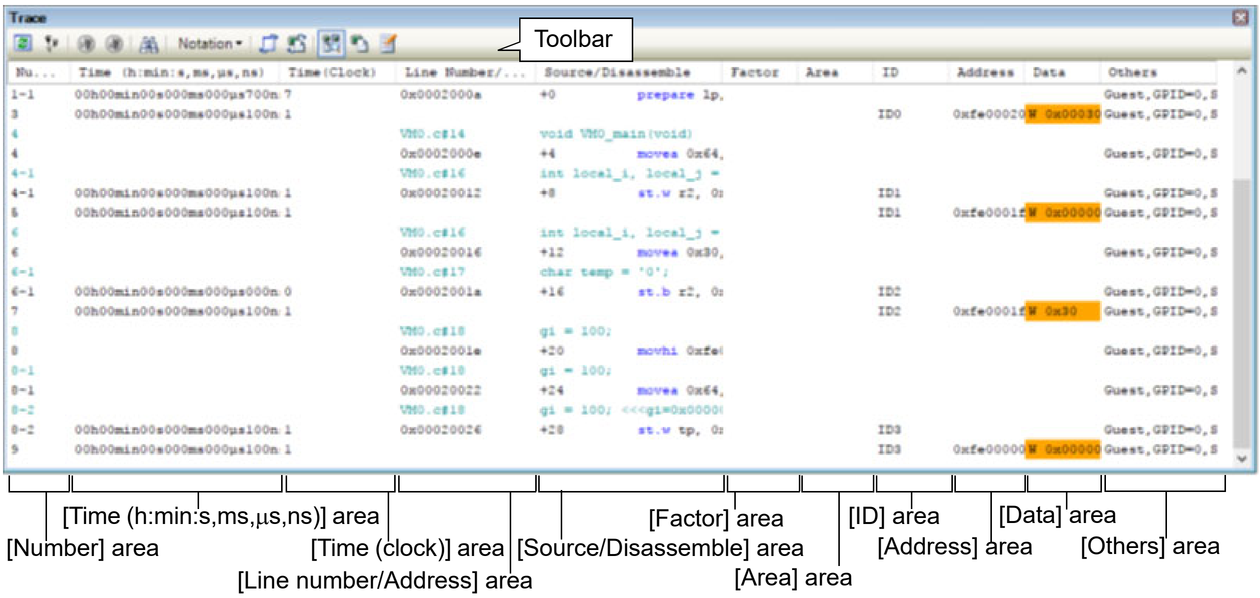
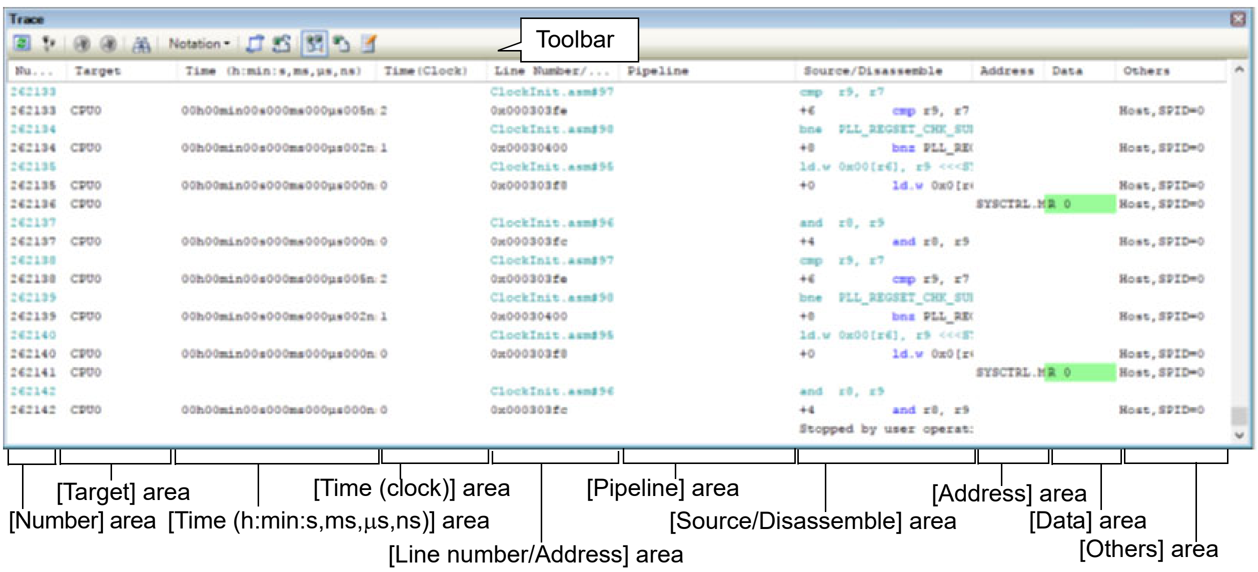


Figure 2.71 Display Trace Data (Trace Panel [Simulator])



This section described the following.

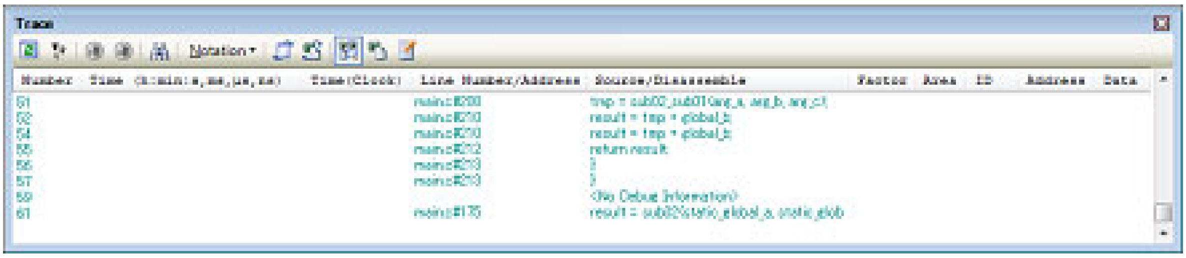
2.14.6.1 Change display mode

Display mode can be changed to the purpose when clicking the buttons below in the toolbar.
Note that these buttons are disabled while the tracer is running.

Table 2.15 Display Modes of Trace Panel

Button	Display Mode	Displayed Content
	Mixed display mode	Displays the instruction (disassemble results), labels, source text (corresponding source line), point trace results, and break causes (default).
	Disassemble display mode	Displays the instruction (disassemble results), labels, point trace results, and break causes.
	Source display mode	Displays the source text (corresponding source line), and break causes. However, when a place where no debugging information is present is executed, "<No Debug Information>" is displayed.

Figure 2.72 Example of Source Display Mode View (Trace Panel)



2.14.6.2 Change display format of values

The display format of the [Line Number/Address], [Address] and [Data] area can be changed using buttons below on the toolbar.
Note that these buttons are disabled while the tracer is running.

Notation	The following buttons to change the notation of a data value are displayed.
Hexadecimal	Displays values on this panel in hexadecimal number (default).
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.

2.14.6.3 Link with other panels

Items in the trace panel can be linked to other panels using the currently selected line address as a pointer (window focus will not move).
Click the button on the toolbar to start linking to the Editor panel. Click the button on the toolbar to start linking to the Disassemble panel.
If the button is clicked again, the link is disconnected.

Remark The Editor panel/Disassemble panel opens when selecting the [Jump to Source]/[Jump to Disassemble] from the context menu with moving the caret to the source line/address corresponding to the address of the currently selected line (focus is moved).

2.14.7 Clear the trace memory

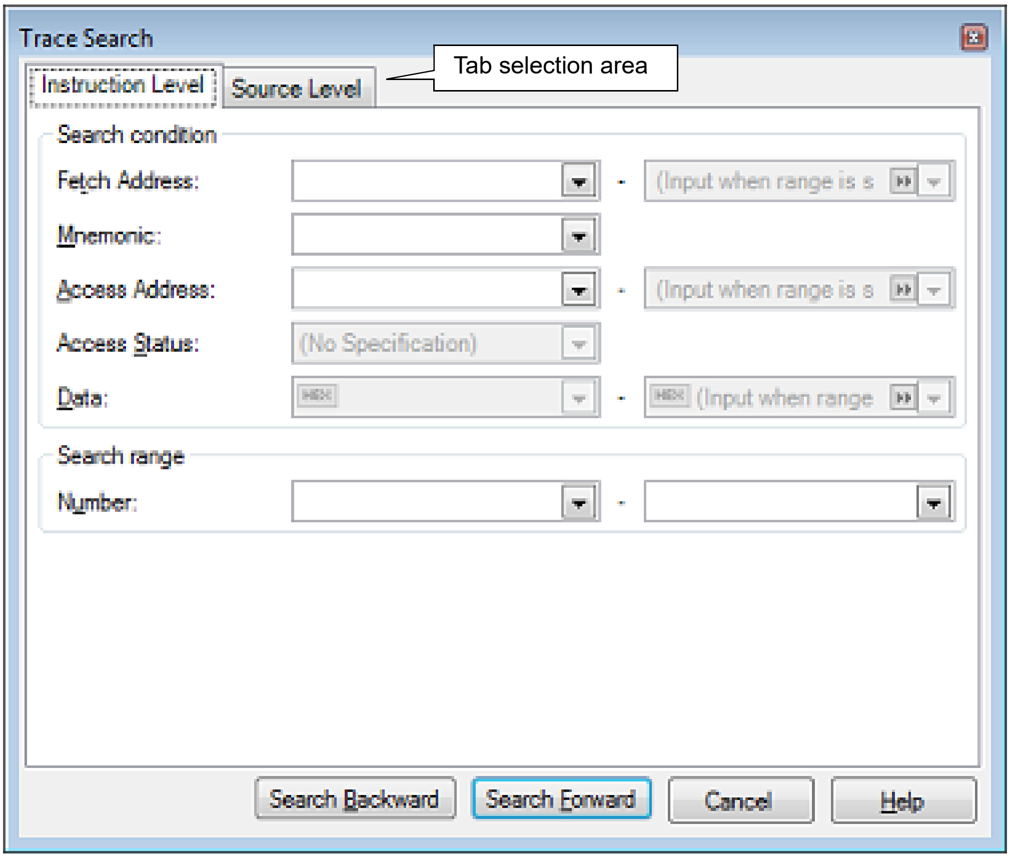
To clear the collected trace data contents, click the button on the toolbar.
Note that this button is disabled while a tracer is running.

Remark When [Yes] is selected in the [Clear trace memory before running] property in the [Trace] category on the [Debug Tool Settings] tab of the Property panel, the trace memory is cleared each time a program is executed.

2.14.8 Search the trace data

To search for the collected trace data, click the button to open the Trace Search dialog box (note that the search is disabled during execution of a program).
In this dialog box, follow the steps below.
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.

Figure 2.73 Search Trace Data (Trace Search Dialog Box)



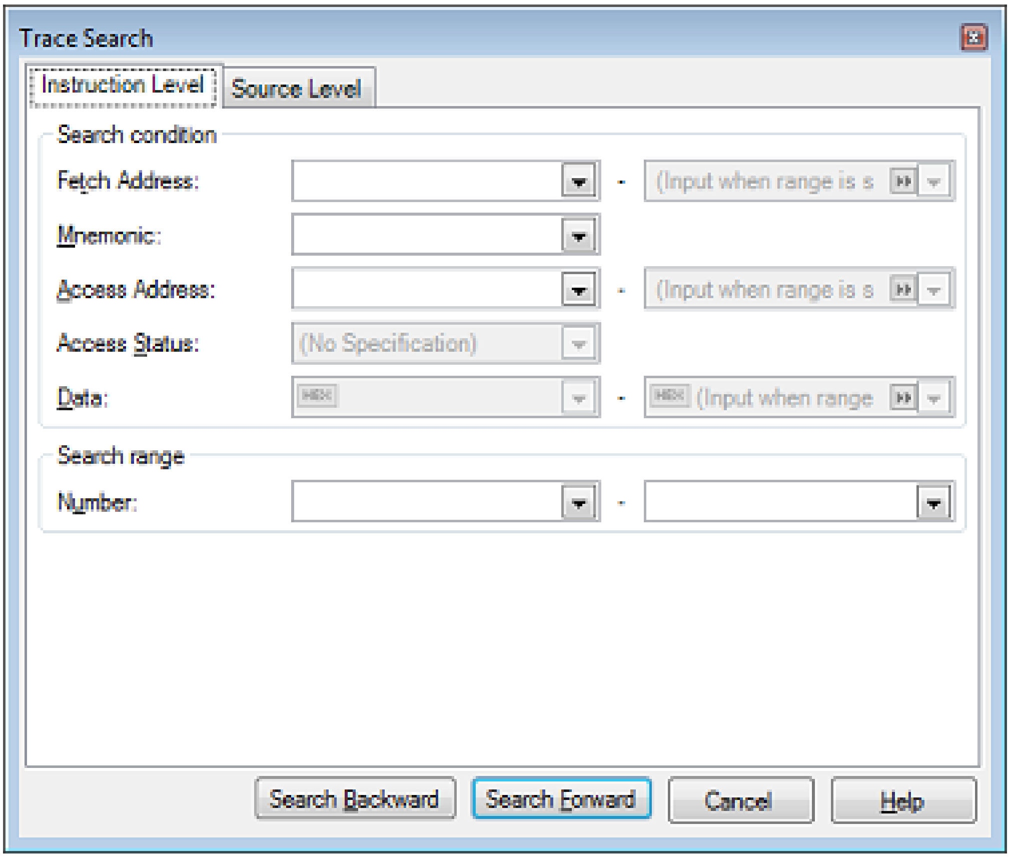
This section describes the following.

2.14.8.1 Search in the instruction level

Search for the trace data at the instruction level.
Select the [Instruction Level] tab and then follow the steps below.

Caution When you search for the trace data at the instruction level, the display mode must be set in the Trace panel to the Mixed display mode or Disassemble display mode.

Figure 2.74 Search Trace Data in Instruction Level



(1) Specify [Fetch Address]
Specify the fetch address if it is a required search parameter.
You can either type address expressions directly into the text boxes, or select it from the input history via the drop-down list (up to 10 items).
The fetch address can also be specified as a range. In this case, specify a range by specifying address expressions in both the left and right text boxes.
If the right-hand text box is blank or contains the text [(Input when range is specified)], then the fixed address specified in the left-hand text box will be searched.
Note that if an address value greater than the microcontroller address space is specified, the upper address value is masked.
An address value greater than the value expressed within 32 bits cannot be specified.

(2) Specify [Mnemonic]
Specify the mnemonic if it is a required search parameter.
The specified character strings in this area is searched within the [Source/Disassemble] area of the Trace panel.
You can either type a mnemonic directly into the text boxes, or select one from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and partial matches are also allowed.

(3) Specify [Access Address]
Specify the access address if it is a required search parameter.
You can either type the address value directly into the text boxes (in hexadecimal number), or select it from the input history via the drop-down list (up to 10 items).
The access address can also be specified as a range. In this case, specify a range by specifying address expressions in both the left and right text boxes.
If the right-hand text box is blank or contains the text [(Input when range is specified)], then the fixed address specified in the left-hand text box will be searched.
Note that if an address value greater than the microcontroller address space is specified, the upper address value is masked.
An address value greater than the value expressed within 32 bits cannot be specified.

(4) Specify [Access Status]
This item is only enable if a value for Specify [Access Address] is specified.
Select the access type (Read/Write, Read, Write, Vector Read and DMA) from drop-down list.

Select [(No Specification)] if you do not wish to limit access types.

(5) Specify [Data]
This item is only enable if a value for Specify [Access Address] is specified.
Specify the access data.
You can either type the data directly into the text boxes (in hexadecimal number), or select it from the input history via the drop-down list (up to 10 items).
The data can also be specified as a range. In this case, specify a range by specifying data in both the left and right text boxes.
If the right-hand text box is blank or contains the text [(Input when range is specified)], then the fixed data specified in the left-hand text box will be searched.

(6) Specify [Number]
Specify the range within the trace data to search via the number displayed in the [Number] area of the Trace panel.
Specify the starting number in the left text box, and the ending number in the right text box ("0" to "*last number*" are specified by default).
You can either type the numbers directly into the text boxes (in base-10 format), or select them from the input history via the drop-down list (up to 10 items).
If the left-hand text box is left blank, it is treated as if "0" were specified.
If the right-hand text box is left blank, it is treated as if the last number were specified.

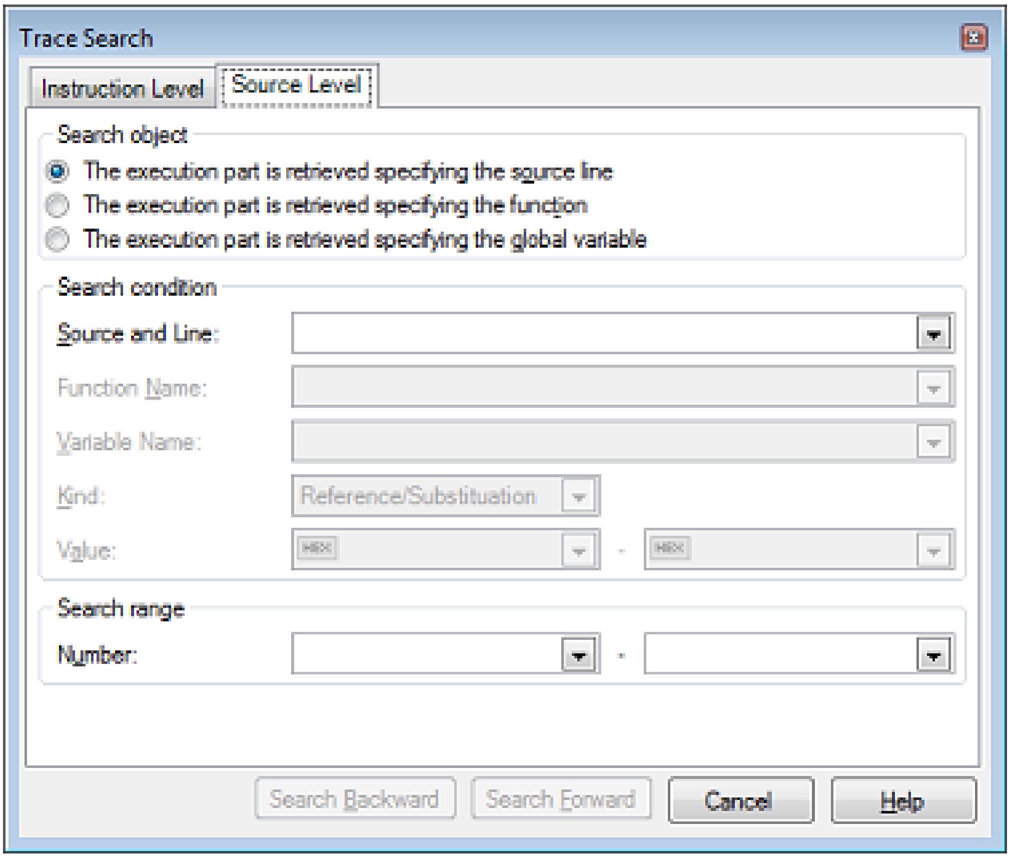
(7) Click the [Search Backward]/[Search Forward] button
When the [Search Backward] button is clicked, search is taken place in the order from the large number to small and the search results are shown selected in the Trace panel.
When the [Search Backward] button is clicked, search is taken place in the order from the small number to large and the search results are shown selected in the Trace panel.

2.14.8.2 Search in the source level

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

Caution When you search for the trace data at the source level, the display mode must be set in the Trace panel to the Mixed display mode or Source display mode.

Figure 2.75 Search Trace Data in Source Level



(1) Search with specifying the source line (default)
Select the [The execution part is retrieved specifying the source line] item in the [Search object] area and then follow the operation below.

(a) Specify [Source and Line]
The specified character strings in this area is searched within the [Line Number/Address] area of the Trace panel.
You can either type the character strings of the source line to be find directly into the text box, or select them from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and partial matches are also allowed.

Example 1. main.c#40

Example 2. main.c

Example 3. main

(b) Specify [Number]
Specify the range within the trace data to search via the number displayed in the [Number] area of the Trace panel.
Specify the starting number in the left text box, and the ending number in the right text box ("0" to "l*ast number*" are specified by default).
You can either type the numbers directly into the text boxes (in base-10 format), or select them from the input history via the drop-down list (up to 10 items).
If the left-hand text box is left blank, it is treated as if "0" were specified.
If the right-hand text box is left blank, it is treated as if the last number were specified.

(c) Click the [Search Backward]/[Search Forward] button
When the [Search Backward] button is clicked, search is taken place in the order from the large number to small and the search results are shown selected in the Trace panel.
When the [Search Backward] button is clicked, search is taken place in the order from the small number to large and the search results are shown selected in the Trace panel.

(2) Search with specifying the function name
Select the [The execution part is retrieved specifying the function] item in the [Search object] area and then follow the operation below.

(a) Specify [Function Name]
You can either type the function name to be find directly into the text box, or select it from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and only complete matches are retrieved.

(b) Specify [Number]
Specify the range within the trace data to search via the number displayed in the [Number] area of the Trace panel.
Specify the starting number in the left text box, and the ending number in the right text box ("0" to "*last number*" are specified by default).
You can either type the numbers directly into the text boxes (in base-10 format), or select them from the input history via the drop-down list (up to 10 items).
If the left-hand text box is left blank, it is treated as if "0" were specified.
If the right-hand text box is left blank, it is treated as if the last number were specified.

(c) Click the [Search Backward]/[Search Forward] button
When the [Search Backward] button is clicked, search is taken place in the order from the large number to small and the search results are shown selected in the Trace panel.
When the [Search Backward] button is clicked, search is taken place in the order from the small number to large and the search results are shown selected in the Trace panel.

(3) Search with specifying the global variable
Select the [The execution part is retrieved specifying the global variable] item in the [Search object] area and then follow the operation below.

(a) Specify [Variable Name]
You can either type the variable name to be find directly into the text box, or select it from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and only complete matches are retrieved.

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

(c) Specify [Value]
You can either type the accessed variable value directly into the text box, or select one from the input history via the drop-down list (up to 10 items).
The variable value can also be specified as a range. In this case, specify a range by specifying variable values in both the left and right text boxes.
If the right-hand text box is blank, then access locations with the fixed variable values specified in the left-hand text box will be searched for.

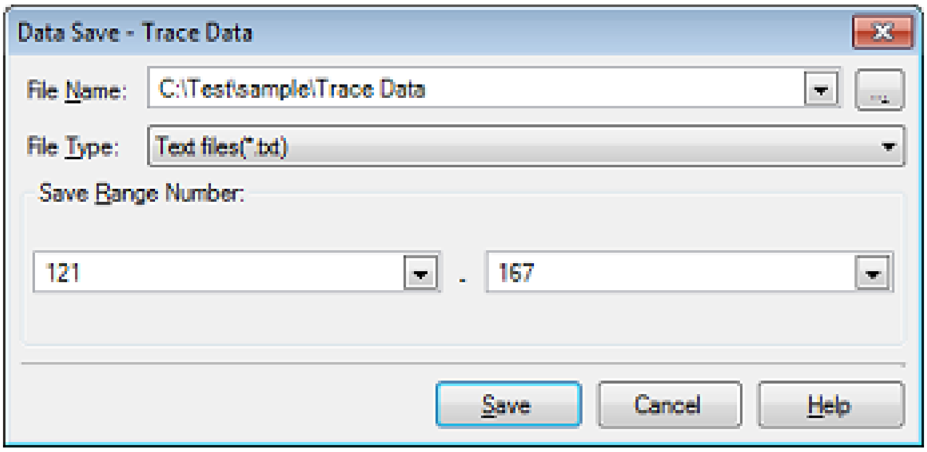
(d) Specify [Number]
Specify the range within the trace data to search via the number displayed in the [Number] area of the Trace panel.
Specify the starting number in the left text box, and the ending number in the right text box ("0" to "*last number*" are specified by default).
You can either type the numbers directly into the text boxes (in base-10 format), or select them from the input history via the drop-down list (up to 10 items).
If the left-hand text box is left blank, it is treated as if "0" were specified.
If the right-hand text box is left blank, it is treated as if the last number were specified.

(e) Click the [Search Backward]/[Search Forward] button
When the [Search Backward] button is clicked, search is taken place in the order from the large number to small and the search results are shown selected in the Trace panel.
When the [Search Backward] button is clicked, search is taken place in the order from the small number to large and the search results are shown selected in the Trace panel.

2.14.9 Save the contents of execution history

Contents of the collected trace data can be saved with range selection in text files (*.txt)/CSV files (*.csv).
When saving to the file, the latest information is acquired from the debug tool, and it is saved in accordance with the display format on this panel.
The following Data Save dialog box can be opened by selecting the [File] menu >> [Save Trace Data As...].
In this dialog box, follow the steps below.

Figure 2.76 Save Execution History (Data Save Dialog Box)



(1) Specify [File Name]
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 [...] button, and selecting a file via the Select Data Save File dialog box.

(2) Specify [File Type]
Select the format in which to save the file from the following drop-down list.
The following file formats can be selected.

List Item	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) Specify [Save Range Number]
Specify the range of the number to save via "start number" and "end number".
Directly enter decimal number in each text box or select from the input history displayed in the drop-down list (up to 10 items).
When saving all the trace data, select the [All Trace Data] item in the drop-down list at the left (the right text box becomes invalid).
If a range is selected in the panel, that range is specified as the default. If there is no selection, then the range currently visible in the panel is specified.

(4) Click the [Save] button
Trace data is saved in the specified file with the specified format.

Figure 2.77 Output Example of Trace Data

```
Number  Target  Time  Clock  Line/Address  Pipeline  Source/Disassemble  Factor
Area  ID  Address  Data
--------------------------------------------------------------------------------
Number  Target  Time  Clock  Line/Address  Pipeline  Source/Disassemble  Factor
Area  ID  Address  Data
   :      :       :     :        :            :            :                :
  :    :     :       :
```

Remark Items of trace data output differ depending on the debug tool used.

2.14.10 Output information by embedding debug instructions

The RH850 family is equipped with a function for the output of software tracing information in response to debug instructions embedded within the user application.

Instructions embedded within the user application can be used to analyze applications.
As part of the solutions we provide for CS+, the IDE takes advantage of DBTAG instructions, which implement the function for the output of software tracing information.

Remark For details on the individual debug instructions, see "RH850G3M/G3MH/G3K/G3KH User's Manual: Debug Instructions".

(1) Usage of debug instructions in the solutions
For the usage of debug instructions in the solutions, see the following.

- 2.23 Exclusive Control Checking Tool
- 2.25 Debugging CAN Bus Reception Procedures [Full-spec emulator][E1][E20]

(2) Setting the operation of software tracing
See "2.14.1 Configure the trace operation" for setting the operation of software tracing.

(3) The output results of software tracing

[IE850A][Full-spec emulator][E1][E20][E2]
To refer to the output results of external tracing or internal tracing of the IE850A, use the Trace panel or the Python console.

[Simulator]
To refer to the output results of software tracing, use the Python console.

(4) imm10 for DBTAG
For the imm10 values which can be output by the DBTAG instruction, the following format is defined in consideration of using each of the solutions for CS+ at the same time.

[9:3]: ID number
[2:0]: Category

CS+ uses 0b001 as the category.
The following shows the ID numbers used in each of the solutions (values within parentheses are the full representation in the imm10 values).

- Exclusive control checking tool
  0x0(0x1), 0x1(0x9), 0x2(0x11), 0x3(0x19)
- Debugging CAN bus reception procedures
  0x4(0x21), 0x5(0x29), 0x6(0x31), 0x7(0x39), 0x8(0x41), 0x9(0x49), 0xa(0x51), 0xb(0x59), 0xc(0x61), 0xd(0x69)

If you wish to use DBTAG within a user application, we recommend specifying a value other than 0b001 as the category.

2.15 Measure Execution Time of Programs

This section describes how to measure the execution time of the program.

Remark For "measurement of execution time of programs" for a microcontroller that supports multi-core, see also to "2.9 Select a Core (PE)".

2.15.1 Measure execution time until stop of the execution

In the debug tool, there is a function to measure the program execution time (Run-Break time) from the start to the stop. Therefore, when the program starts its execution, the execution time is automatically measured.
You can check the result of the measurement by either one of the following.

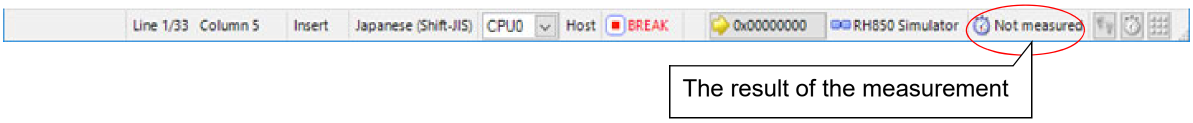
Caution 1. The execution time cannot be measured when Step In or Step Over is performed.

Caution 2. [Simulator]
To measure the Run-Break time, [Yes] must be specified with the [Use timer function] property in the [Timer] category on the [Debug Tool Settings] tab of the Property panel.

Remark This function is operated by a Run-Break Timer event, which is one of the built-in events set by default in the debug tool.

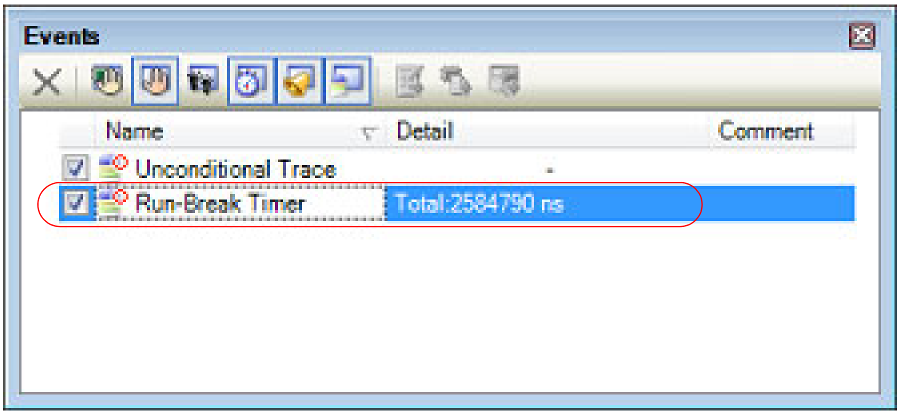
(1) Check in the status bar
After the program is stopped, the result of the measurement is displayed in the status bar on the Main window (when measurements have not been performed yet, "Not measured" is displayed).

Figure 2.78 Example of Result of Run-Break Timer Event (Status Bar)



(2) Check on the Events panel
After the program is stopped, the result of the measurement is displayed in the Events panel opened by selecting the [View] menu >> [Event], in event type as "Run-Break Timer".

Figure 2.79 Example of Result of Run-Break Timer Event (Events Panel)



2.15.2 Measure execution time in a section

In the program execution process, the execution time in a section can be measured by setting Timer Result event.
To use this function, follow the procedure described below.

2.15.2.1 Set a Timer Result event
2.15.2.2 Execute the program
2.15.2.3 Edit a Timer Result event [Full-spec emulator][E1][E20]
2.15.2.4 Delete a Timer Result event

Caution 1. Also see "2.19.7 Notes for setting events" for details on Timer Result events (e.g. limits on the number of enabled events).

Caution 2. To use this function, [Yes] must be specified with the [Use timer function] property in the [Timer] category on the [Debug Tool Settings] tab of the Property panel.

2.15.2.1 Set a Timer Result event

To set a Timer Result event, set a timer start event and a timer end event that starts/stops a timer measurement.
Use one of the following methods to set a timer start event and a timer end event.

(1) For execution-related events
The execution time of a desired section can be measured by setting execution-related events as timer start and end events.
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, in accordance with your desired event type, after moving the caret to the target line that has a valid address.

Event Type	Operation
Timer start	Select [Timer Settings] >> [Start Timer] >> [Set Timer *n*[Note]]
Timer end	Select [Timer Settings] >> [Stop Timer] >> [Set Timer *n*[Note]]

Note [Simulator]
Select the channel number (*n*: 1 to 8) in which a Timer Result event is set.
[Full-spec emulator][E1][E20]
Select the channel number (*n*: 1 to 3) in which a Timer Result event is set for each core.

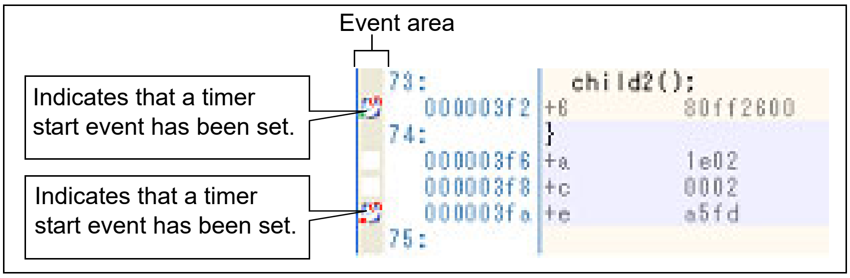
Caution The time for a timer end event will not included in the measurement results. For this reason, set a timer end event to one line below the range for which you wish to measure the run time.

A timer start event or a timer end event is set to the instruction at the start address corresponding to the line of the caret position.
Once a timer start event or a timer end event is set, the following event mark is displayed in the event area of the line/address that an event is set.

Table 2.16 Event Marks of Timer Start Event/Timer End Event

Event Type	Event Mark
Timer start	
Timer end	

Figure 2.80 Timer Start and Timer End Events Setting Example (Disassemble Panel)



(2) For access-related events
In this product version, this function is not supported.

When a timer start event and a timer end event are set, they are managed collectively on the Events panel as one instance of a Timer Result event (see "2.19 Manage Events"). When you click the "+" mark at a Timer Result event item, detailed information on the timer start event and the timer end event you have set is displayed.

Figure 2.81 Example of Timer Start and Timer End Events (Execution Type) in Events Panel [Simulator]

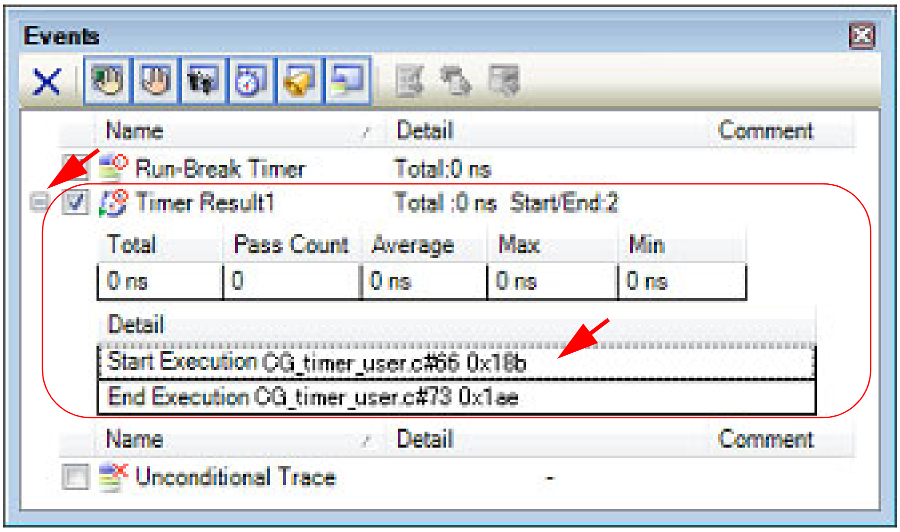
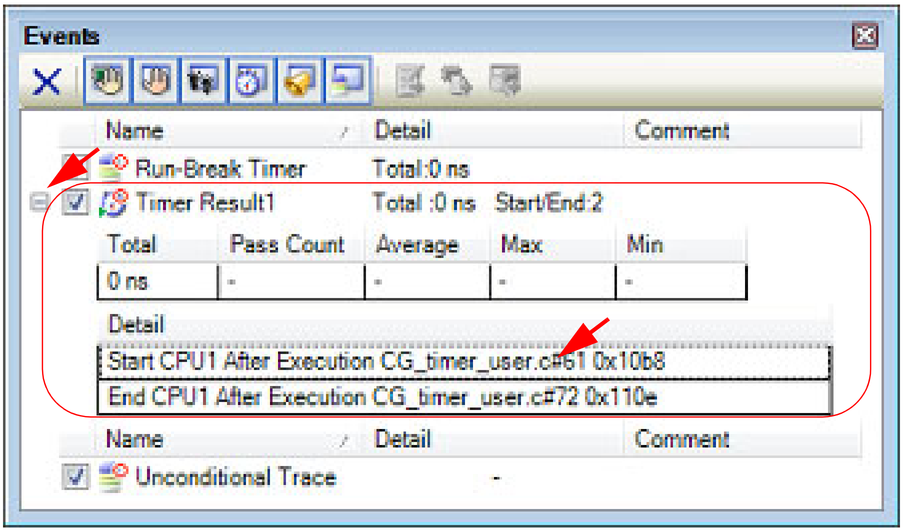


Figure 2.82 Example of Timer Start and Timer End Events (Execution Type) in Events Panel [Full-spec emulator][E1][E20]



Remark Event marks differ depending on the state of set events (see "2.19.1 Change the state of set events (valid/ invalid)").
When an event is set at the point which other event is already set, the event mark ( ) is displayed meaning more than one event is set at the point.

Caution 1. [Full-spec emulator][E1][E20]
Timer measurement can be performed with even only one setting: a timer start event or a timer end event. When only a timer start event is set, timer measurement is terminated when program execution stops. When only a timer end event is set, timer measurement is started when program execution starts.

Caution 2. [Simulator]
Timer measurement is enabled when both a timer start event and a timer end event have been set. Therefore, timer measurement cannot be performed with only one setting: a timer start event or a timer end event.

2.15.2.2 Execute the program

Execute the program (see "2.10 Execute Programs").

When an instruction for which a timer start event or a timer end event has been set is executed, a timer measurement is started or finished.

After the program is stopped, the result of the measurement is displayed in the Events panel opened by selecting the [View] menu >> [Event], in event type as "Timer Result".

This Timer Result 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 2.83 Example of Result of Timer Result Event (Timer Start Event/Timer End Event) [Simulator]

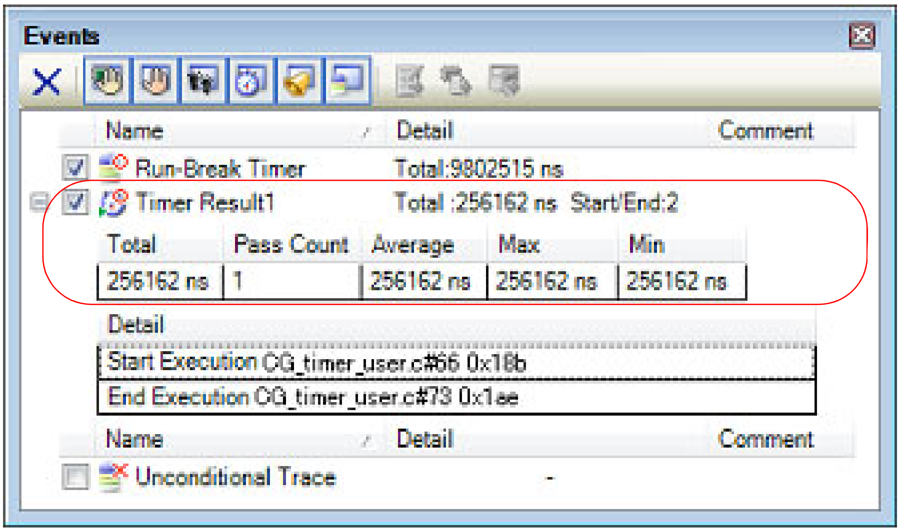
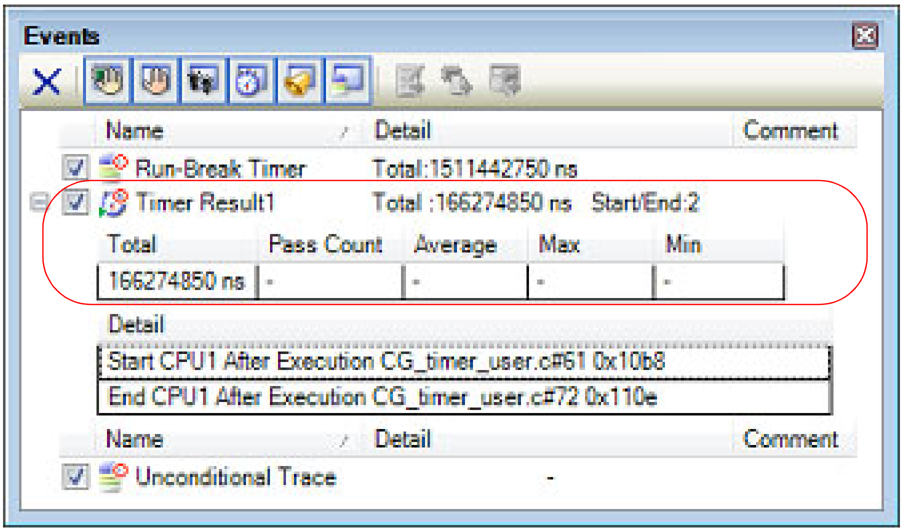


Figure 2.84 Example of Result of Timer Result Event (Timer Start Event/Timer End Event) [Full-spec emulator][E1][E20]



2.15.2.3 Edit a Timer Result event [Full-spec emulator][E1][E20]

Detailed information on a Timer Result event is edited in the Detailed Settings of Timer Measurement dialog box [Full-spec emulator][E1][E20]. This dialog box is opened by selecting the Timer Result event you desire to edit on the Events panel then selecting [Edit Condition...] from the context menu.

In the Detailed Settings of Timer Measurement dialog box [Full-spec emulator][E1][E20], the following can be specified as measurement items.

Only a single item can be measured, and the measurement result for only the selected item is displayed on the Events panel.

Measurement Item	Function
Total Count	Measures the total execution time of the specified section (default).
Max Count	Measures the maximum execution time of the specified section.
Min Count	Measures the minimum execution time of the specified section.

Measurement Item	Function
Pass Count	Measures the pass count of the specified section. This cannot be measured when an end condition has not been set.

2.15.2.4 Delete a Timer Result event

To delete a Timer Result event you have set, on the Editor panel/Disassemble panel, right-click the event mark in the event area and select [Delete Event] from the context menu that is displayed.

Also, there is another way to delete a set event. Select the Timer Result event you want to delete on the Events panel, and then click the button in the toolbar (see "2.19.5 Delete events").

Caution [Simulator]
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.

2.15.3 Measurable time

[Full-spec emulator][E1][E20]
The LPD clock is used for time measurement by a Run-Break Timer event (see "2.15.1 Measure execution time until stop of the execution" for details) or a Timer Result event (see "2.15.2 Measure execution time in a section" for details).
The measurable time is as follows:

Table 2.17 Measurable Time [Full-spec emulator]

	Run-Break Timer Event / Timer Result Event
resolution	Approx. 50 ns (20 MHz)
Measurable time	Approx. 3 min. 30 s (20 MHz) Overflow detection included

Table 2.18 Measurable Time [E1][E20]

	Run-Break Timer Event / Timer Result Event
Max resolution	[E1][E20] Approx. 60 ns (16.5 MHz, LPD: 4-pin) [E2][IE850A] Approx. 30 ns (33 MHz, LPD: 4-pin)
Max measurable time	Approx. 13 min (5.5 MHz, LPD: 4-pin) Overflow detection included

[Simulator]
The CPU clock is used for time measurement by a Run-Break Timer event (see "2.15.1 Measure execution time until stop of the execution" for details) or a Timer Result event (see "2.15.2 Measure execution time in a section" for details).

2.16 Measure Performance [Full-spec emulator][E1][E20]

This section describes how to measure the performance of the program.

Remark For "measurement of performance of programs" for a microcontroller that supports multi-core, see also to "2.9 Select a Core (PE)".

2.16.1 Measure the performance in a section

In the program execution process, the performance in a section can be measured by setting a Performance Measurement event.
To use this function, follow the procedure described below.

2.16.1.1 Set a Performance Measurement event
2.16.1.2 Execute the program
2.16.1.3 Edit a Performance Measurement event
2.16.1.4 Delete a Performance Measurement event

Caution Also see "2.19.7 Notes for setting events" for details on Performance Measurement events (e.g. limits on the number of enabled events).

2.16.1.1 Set a Performance Measurement event

To set a Performance Measurement event, set a performance measurement start event and a performance measurement end event that starts/stops a performance measurement.
Use one of the following methods to set a performance measurement start event and a performance measurement end event.

(1) For execution-related events
The performance of a desired section can be measured by setting execution-related events as performance start and end events.
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, in accordance with your desired event type, after moving the caret to the target line that has a valid address.

Event Type	Operation
Performance measurement start	Select [Performance Measurement Settings] >> [Start Performance Measurement] >> [Set Performance Measurement *n*[Note]]
Performance measurement end	Select [Performance Measurement Settings] >> [Stop Performance Measurement] >> [Set Performance Measurement *n*[Note]]

Note Select the channel number (*n*: 1 to 4) in which a Performance Measurement event is set for each core.

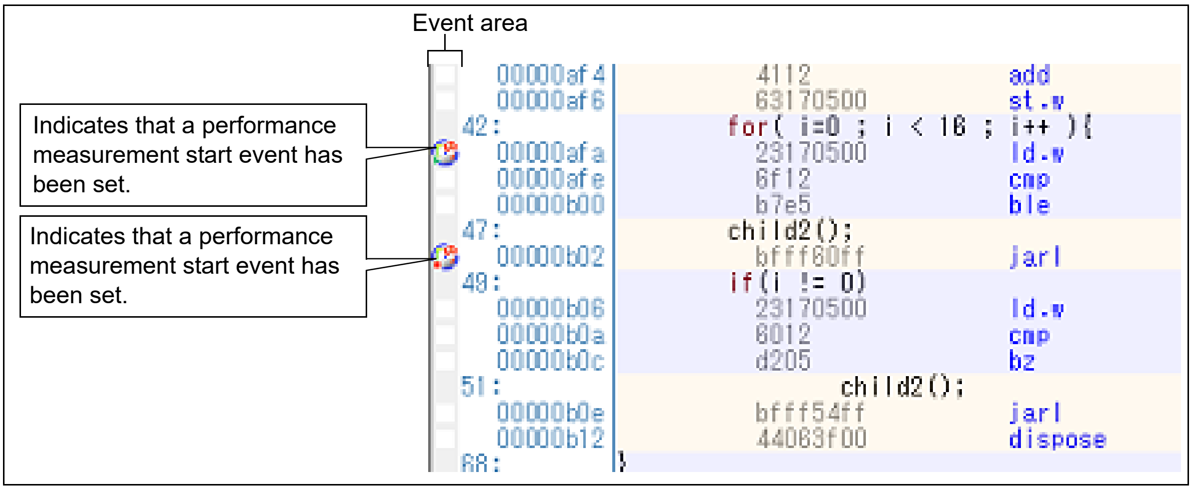
Caution The performance for a performance measurement end event will not included in the measurement results. For this reason, set a performance measurement end event to one line below the range for which you wish to measure the performance.

A performance measurement start event or a performance measurement end event is set to the instruction at the start address corresponding to the line of the caret position.
Once a performance measurement start event or a performance measurement end event is set, the following event mark is displayed in the event area of the line/address that an event is set.

Table 2.19 Event Marks of Performance Measurement Start Event/Performance Measurement End Event

Event Type	Event Mark
Performance Measurement start	
Performance Measurement end	

Figure 2.85 Performance Measurement Start and Performance Measurement End Events Setting Example (Disassemble Panel)



(2) For access-related events
By setting access-related events for a performance measurement start event and a performance measurement end event, it is possible to start and stop performance measurement when a specified access is made to any variable or IOR.

(a) For access to a variable or I/O register on the Editor panel/Disassemble panel
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, in accordance with your desired event type, after moving the caret to the target line that has a valid address.
Note, however, that only global variables, static variables inside functions, and file-internal static variables can be used.

Event Type	Access Type	Operation
Performance measurement start	Read	Select [Performance Measurement Settings] >> [Set Performance Measurement Start Read Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Write	Select [Performance Measurement Settings] >> [Set Performance Measurement Start Write Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Read/Write	Select [Performance Measurement Settings] >> [Set Performance Measurement Start R/W Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
Performance measurement end	Read	Select [Performance Measurement Settings] >> [Set Performance Measurement End Read Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Write	Select [Performance Measurement Settings] >> [Set Performance Measurement End Write Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Read/Write	Select [Performance Measurement Settings] >> [Set Performance Measurement End R/W Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.

At this time, if you have specified a value in the text box in the context menu, performance measurement is started or finished only when a read/write is performed with a specified value. If no value is specified, performance measurement is started or finished when a read/write is performed to the selected variable or IOR, regardless of the value.

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

Caution 2. Variables or IOR at lines that have no valid addresses cannot be used for performance measurement start events and performance measurement end events.

(b) For access to a registered watch-expression
Perform this operation in the Watch panel.
Select the watch-expression as the subject to access and perform the following operation from the context menu.
Note, however, that only global variables, static variables inside functions, file-internal static variables, and I/O register can be used.

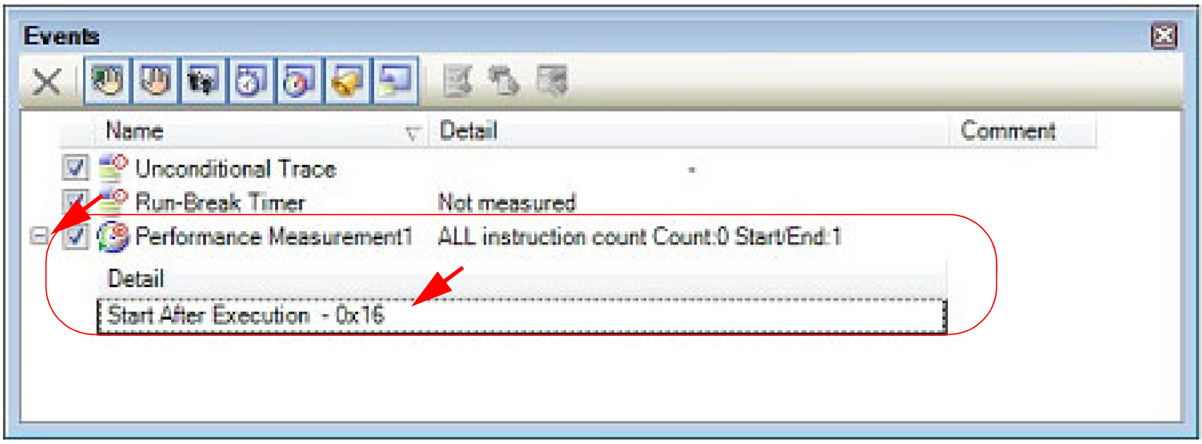
Event Type	Access Type	Operation
Performance measurement start	Read	Select [Performance Measurement Settings] >> [Set Performance Measurement Start Read Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Write	Select [Performance Measurement Settings] >> [Set Performance Measurement Start Write Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Read/Write	Select [Performance Measurement Settings] >> [Set Performance Measurement Start R/W Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
Performance measurement end	Read	Select [Performance Measurement Settings] >> [Set Performance Measurement End Read Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Write	Select [Performance Measurement Settings] >> [Set Performance Measurement End Write Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.
	Read/Write	Select [Performance Measurement Settings] >> [Set Performance Measurement End R/W Value] >> [Set Performance Measurement *n*], and then press the [Enter] key.

At this time, if you have specified a value in the text box in the context menu, performance measurement is started or finished only when a read/write is performed with a specified watch-expression. If no value is specified, performance measurement is started or finished when a read/write is performed to the selected variable or IOR, regardless of the value.

Caution 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.

When a performance measurement start event and a performance measurement end event are set, they are managed collectively on the Events panel as one instance of a Performance Measurement event (see "2.19 Manage Events"). When you click the "+" mark at a Performance Measurement event item, detailed information on the performance measurement start event and the performance measurement end event you have set is displayed.

Figure 2.86 Example of Performance Measurement Start and Performance Measurement End Events (Execution Type) in Events Panel



Remark Event marks differ depending on the event state (see "2.19.1 Change the state of set events (valid/ invalid)").
When an event is set at the point which other event is already set, the event mark () is displayed meaning more than one event is set at the point.

Caution Performance measurement can be performed with even only one setting: a performance measurement start event or a performance measurement end event. When only a performance measurement start event is set, performance measurement is terminated when program execution stops. When only a performance measurement end event is set, performance measurement is started when program execution starts.

2.16.1.2 Execute the program

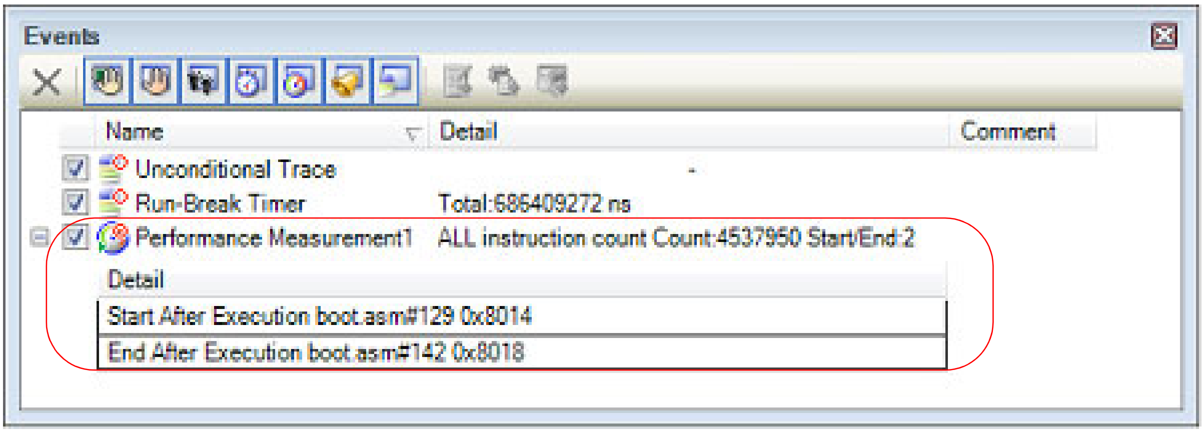
Execute the program (see "2.10 Execute Programs").

When an instruction for which a performance measurement start event or a performance measurement end event has been set is executed, a performance measurement is started or finished.

After the program is stopped, the result of the measurement is displayed in the Events panel opened by selecting the [View] menu >> [Event], in event type as "Performance Measurement Result".

This Performance Measurement Result is a particular type of event that is displayed on only the Events panel when either a performance measurement start event or a performance measurement end event has been set.

Figure 2.87 Example of Result of Performance Measurement Event (Performance Measurement Start Event/ Performance Measurement End Event)



2.16.1.3 Edit a Performance Measurement event

Detailed information on a Performance Measurement event is edited in the Detailed Settings of Performance Measurement dialog box [Full-spec emulator][E1][E20]. This dialog box is opened by selecting the Performance Measurement event you desire to edit on the Events panel then selecting [Edit Condition...] from the context menu.

In the Detailed Settings of Performance Measurement dialog box [Full-spec emulator][E1][E20], the following can be specified as measurement modes and measurement items.

Only a single item can be measured for each, and the measurement result for the selected item is displayed on the Events panel.

Measurement Mode	Function
Total Count	Measures the total count of the measurement item of the specified section (default).
Max Count	Measures the maximum count of the measurement item of the specified section.
Min Count	Measures the minimum count of the measurement item of the specified section.
New Count	Measures the new count of the measurement item of the specified section.
Pass Count	Measures the pass count of the specified section. This cannot be measured when an end condition has not been set. When "Pass Count" is selected, the current value of the [Measurement item] property is ignored and the pass count value is measured.

[RH850G3M, RH850G3K, RH850G3MH, RH850G3KH]

Measurement Item	Function
ALL instruction count	Measures the number of times any instructions in the specified section are executed (default).
Branch instruction count	Measures the number of times any instructions that trigger branching in the specified section are executed.
EI level interrupt count	Measures the number of times EI-level interrupts in the specified section are accepted.
FE level interrupt count	Measures the number of times FE-level interrupts in the specified section are accepted.
ALL instruction async exception count	Measures the number of times any instruction async exceptions in the specified section are accepted.
ALL instruction sync exception count	Measures the number of times any instruction sync exceptions in the specified section are accepted.
Clock cycle	Measures the number of clock cycles in the specified section.
Non-interrupt cycle	Measures the number of cycles excluding the interrupt processing in the specified section.
Interrupt disable cycle of DI/EI	Measures the number of cycles in which DI/EI interrupts are disabled in the specified section.
CPU issued instruction fetch request count	Measures the number of instruction fetch requests issued by CPU in the specified section.
Response count for CPU issued instruction fetch request	Measures the number of instruction cache non-wait responses for instruction fetch requests issued by CPU in the specified section.
Flash ROM data request count	Measures the number of flash ROM data requests in the specified section.

[RH850G4MH, RH850G4KH]

Measurement Item	Function
ALL instruction count	Measures the number of times any instructions in the specified section are executed (default).
Branch instruction count (excluding condition mismatch Bcond instruction, Loop instruction, exception instruction)	Measures the number of times branch instruction (excluding condition mismatch Bcond instruction, Loop instruction, exception instruction) in the specified section are accepted.
Conditional branch instruction count (Bcond instruction, Loop instruction)	Measures the number of times conditional branch instruction (Bcond instruction, Loop instruction) in the specified section are accepted.
Branch prediction misses of conditional branch instruction count (Bcond instruction, Loop instruction)	Measures the number of times branch prediction misses of conditional branch instruction (Bcond instruction, Loop instruction) in the specified section are accepted.
EI level interrupt count	Measures the number of times EI-level interrupts in the specified section are accepted.
FE level interrupt count	Measures the number of times FE-level interrupts in the specified section are accepted.
ALL instruction async exception count	Measures the number of times any instruction async exceptions in the specified section are accepted.
ALL instruction sync exception count	Measures the number of times any instruction sync exceptions in the specified section are accepted.
Stall cycles issued to the instruction execution unit	Measures the number of stall cycles issued to the instruction execution unit in the specified section.
Clock cycle	Measures the number of clock cycles in the specified section.
Non-interrupt cycle	Measures the number of cycles excluding the interrupt processing in the specified section.
Interrupt disable cycle of DI/EI	Measures the number of cycles in which DI/EI interrupts are disabled in the specified section.
CPU issued instruction fetch request count	Measures the number of instruction fetch requests issued by CPU in the specified section.
Response count for CPU issued instruction fetch request	Measures the number of instruction cache non-wait responses for instruction fetch requests issued by CPU in the specified section.
Background interrupt count	Measures the number of times interrupts in the specified section are accepted in the background.
Background EI level interrupt count	Measures the number of times EI-level interrupts in the specified section is accepted in the background.
Background FE level interrupt count	Measures the number of times FE-level interrupts in the specified section is accepted in the background.
Background instruction async exception count	Measures the number of times any instruction async exceptions in the specified section are accepted in the background.

2.16.1.4 Delete a Performance Measurement event

To delete a Performance Measurement event you have set, on the Editor panel/Disassemble panel, right-click the event mark in the event area and select [Delete Event] from the context menu that is displayed.

Also, there is another way to delete a set event. Select the Performance Measurement event you want to delete on the Events panel, and then click the button in the toolbar (see "2.19.5 Delete events").

2.16.2 Measurable range

The CPU clock is used for performance measurement by a Performance Measurement event (see "2.16.1 Measure the performance in a section" for details).

If "Clock cycle", "Non-interrupt cycle", or "Interrupt disable cycle of DI/EI" is set as a measurement item (see "Detailed Settings of Performance Measurement dialog box [Full-spec emulator][E1][E20]"), the number of CPU cycles is measured. If any other item is set as a measurement item, the number of counts is measured. The measurable range is from 0 to 4294967295.

Example When the CPU clock frequency is 320 MHz and the measurement result is 10 cycles, the result is 31.25 nanoseconds (ns) when converted into time.

2.17 Measure Coverage [Simulator]

This section describes coverage measurements that are conducted using the coverage function.

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:

- 1 MByte space of addresses 0x000000 to 0x0FFFFF in the internal ROM area (fixed measurement area)
- Any 1 MByte space other than the fixed measurement area above (see "2.17.1 Configure the coverage measurement")

Remark 1. C0 coverage: Instruction coverage (statement coverage)
For example, if all instructions (statements) in code are executed at least once, then C0 = 100%.

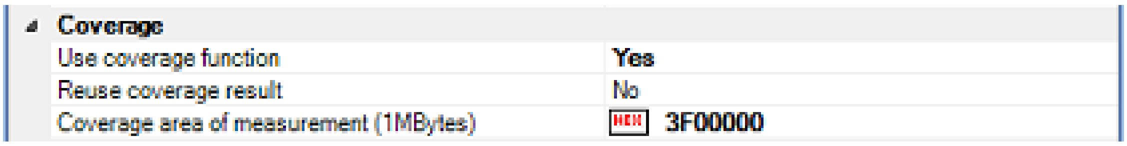
Remark 2. For "coverage measurements" for a microcontroller that supports multi-core, see also to "2.9 Select a Core (PE)".

2.17.1 Configure the coverage measurement

You need to configure the code coverage measurement before using the coverage function.

You can configure the coverage measurement function in the [Coverage] category on the [Debug Tool Settings] tab of the Property panel as follows:

Figure 2.88 [Coverage] Category



(1) [Use coverage function]
Select whether to use the coverage function.
Select [Yes] to use the coverage function (default: [No]).

(2) [Reuse coverage result]
This property appears only when the [Use coverage function] property is set to [Yes].
The currently obtained results of code coverage measurements are automatically saved when CS+ is disconnected from the debug tool. The next time it is connected to the debug tool, specify whether or not you want to reproduce the contents of saved measurement results.
Select [Yes] to reproduce the contents of previously obtained code coverage measurement results (default: [No]).

Caution This function applies to only the internal ROM area.

(3) [Coverage area of measurement(1MBytes)]
This property appears only when the [Use coverage function] property is set to [Yes].
Specify the code coverage measurement area.
Directly enter the start address of any 1 Mbyte space other than the internal ROM area (0x000000 - 0x0FFFFF) in hexadecimal number (default: [100000]).

2.17.2 Display the coverage measurement result

When the program starts running, a coverage measurement is automatically begun, and when the program stops running, the coverage measurement is terminated at the same time.

(1) Code coverage rates

(a) Display of code coverage rates for source text lines and disassembled text lines
The code coverage rates are indicated on the Editor panel/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 2.21") according to their code coverage rate that was calculated based on the formula described in "Table 2.20".
Note that the results are not shown when disconnected from the debug tool or during the program execution.
Selecting [Clear coverage information] from the context menu in the Editor panel/Disassemble panel will reset all the coverage information acquired, including the color-coded display on each panel.

Caution When the selected microcontroller supports multi-core, in the Local RAM self area, note that the measurement results will be displayed only for the access in the currently selected PE (see "2.9 Select a Core (PE)").

Table 2.20 Method for Calculating Code Coverage Rates for Source Lines and Disassemble Lines

Panel	Calculation Method
Editor panel	"Number of bytes of code executed in the address range corresponding to the source text line" / "Total number of bytes of code in the address range corresponding to the source text line"
Disassemble panel	"Number of bytes of code executed in the address range corresponding to the disassembled text line" / "Total number of bytes of code in the address range corresponding to the disassembled text line"

Table 2.21 View of Code Coverage Measurement Result (Default)

Code Coverage	Background Color
100%	Source text/disassembled text
1 to 99%	Source text/disassembled text
0% (not yet executed)	Source text/disassembled text

Remark 1. The code coverage measurement result displayed on each panel is automatically updated at every program break.

Remark 2. The above background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box.

Remark 3. The above background colors do not apply to the lines that are outside of the subject area.

Remark 4. If the downloaded lode module file is older than the source file currently being open, the displaying of the code coverage measurement result is not performed in the Editor panel.

Figure 2.89 View of Code Coverage Measurement Result (Editor Panel)

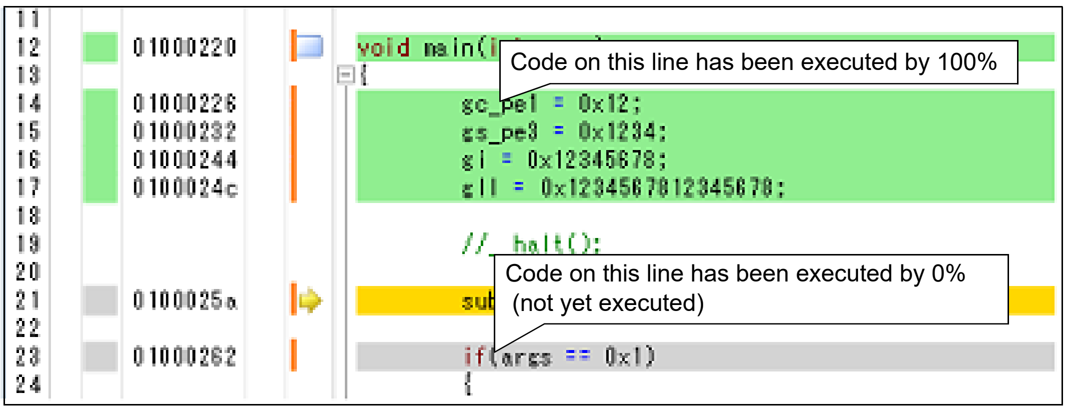
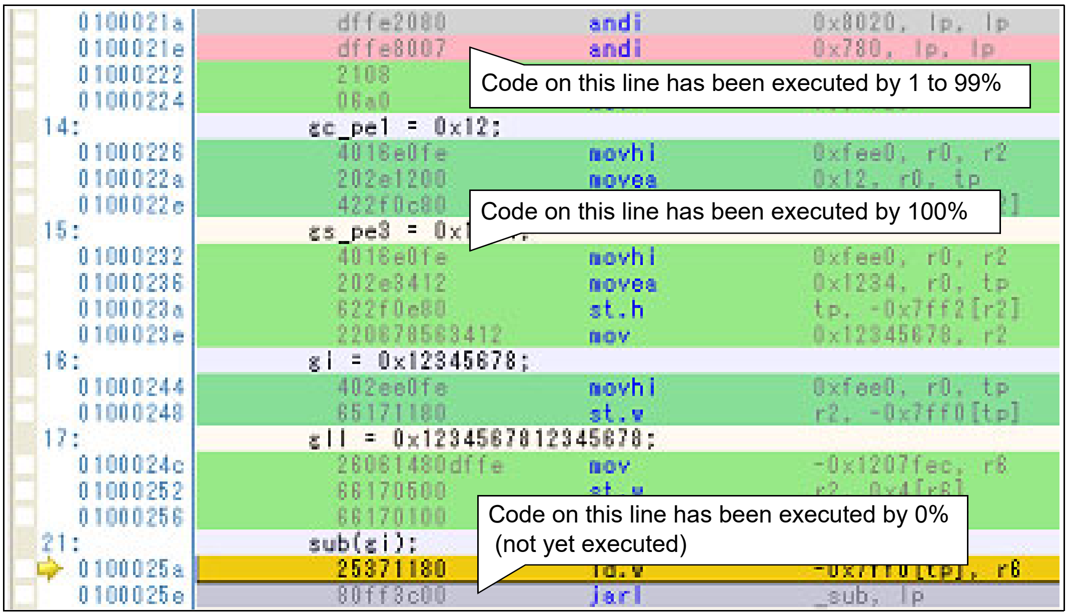


Figure 2.90 View of Code Coverage Measurement Result (Disassemble Panel)



(b) Display of code coverage rates for each function

Code coverage rates of each function can be checked via the [Code Coverage[%]] item in the Function List panel of the analyze tool. For details on "the code coverage rates of the function", see "CS+ Integrated Development Environment User's Manual: Analysis".

(2) Data coverage rates

Data coverage rates of each variable can be checked via the [Data Coverage[%]] item in the Variable List panel of the analyze tool. For details on "the data coverage rates of the variable", see "CS+ Integrated Development Environment User's Manual: Analysis".

2.18 Set an Action into Programs

This section describes how to set the specified action into the program.

2.18.1 Inset printf

By setting a Printf event that is one of "action events", the value of the specified variable expression can be output to the Output panel by executing a printf command after temporarily stopping the program in execution at an arbitrary position.
To use this function, follow the steps below.

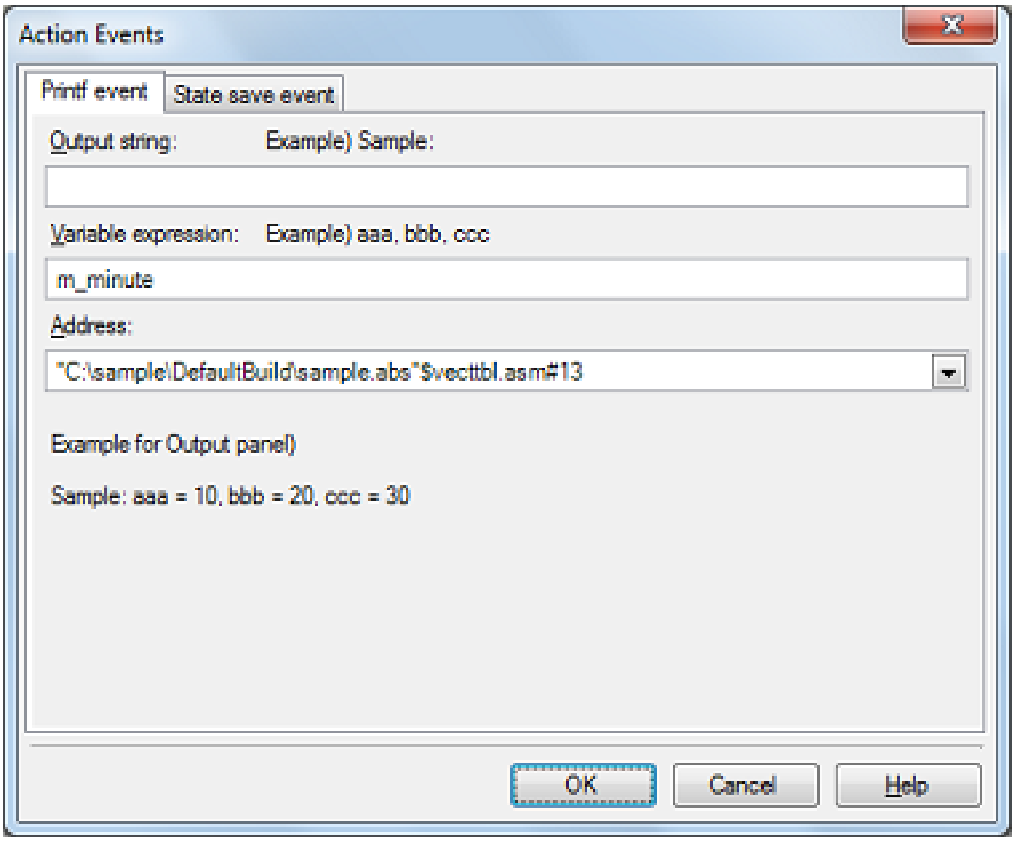
Caution 1. [Full-spec emulator][E1][E20]
Printf event is implemented using the Software break function [Full-spec emulator][E1][E20]. Therefore, you need to select [Yes] in the [Use software break] property in the [Break] category on the [Debug Tool Settings] tab of the Property panel before setting a Printf event.

Caution 2. Also see "2.19.7 Notes for setting events" for details on a Printf event (e.g. limits on the number of enabled events).

Caution 3. No action events occur during step execution (//) or program execution ignoring break-related events ().

(1) Set a Printf event
Set a Printf event to the position where you want to execute the printf command in the Editor panel/Disassemble panel.
On each panel, select [Register Action Event...] from the context menu after moving the caret to the line that has a valid address to open the Action Events dialog box below.
In this dialog box, follow the steps below.

Figure 2.91 Set Printf Event (Action Events Dialog Box: [Printf event] tab)



(a) Specify [Output string]
Directly enter from the keyboard the characters to add when output to the Output panel.
Characters must be in one line (spaces allowed).

(b) Specify [Variable expression]
Specify the variable expression for the Printf event to take place.
Type a variable expression directly into the text box (up to 1024 characters).

You can specify up to 10 variable expressions for a single Printf event by separating them with commas ",".
If this dialog box opens with a variable expression selected in the Editor panel/Disassemble panel, the selected variable expression appears as the default.
For the basic input format that can be specified as variable expressions and the values output by Printf event, see "Table A.12 Relationship between Variable Expressions and Output Value (Printf Event)".

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (see "2.21.2 Symbol name completion function").

(c) Specify [Address]
Specify the address at which to set the Printf event.
The address of the location currently being specified is displayed by default.
If you want to edit this area, 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).

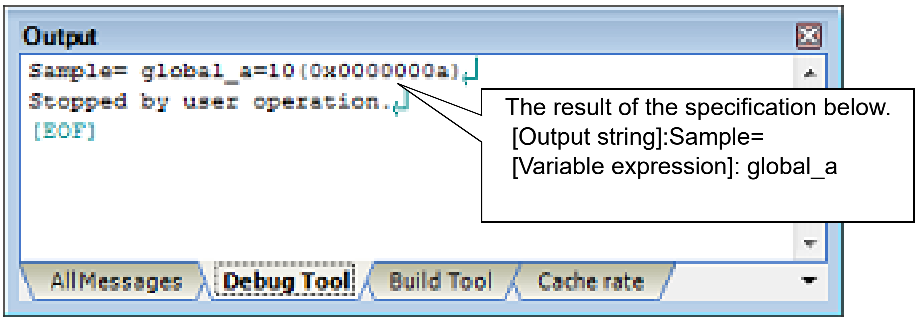
Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (see "2.21.2 Symbol name completion function").

(d) Click the [OK] button
Set the Printf event to the line/address 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 "2.19 Manage Events").

(2) Execute the program
Execute the program (see "2.10 Execute Programs").
By executing the program, the program momentarily stops immediately before executing the instruction at the location where this event is set, and the value of the variable expression specified in this dialog box is output to the Output panel.

(3) Check the output result
The output result format from the Printf event in the [Debug Tool] tab of the Output panel are as follows (see "Figure A.32 Output Result Format of Printf Event"):

Figure 2.92 Example of Output Result of Printf Event



(4) Edit the Printf event
You can edit the Print event that has been set once.
To do this, on the Events panel, select [Edit Condition...] from the context menu after selecting the Printf event to be edited. On the Action Events dialog box opened automatically, edit the items, and then click the [OK] button.

2.19 Manage Events

An event represents a certain status of the target system when debugging such as "Address 0x1000 is fetched" and "Data is written to address 0x2000".

In CS+, these events are used as the action trigger of the debug function such as breakpoint, start/stop the tracing, and start/stop the timer.

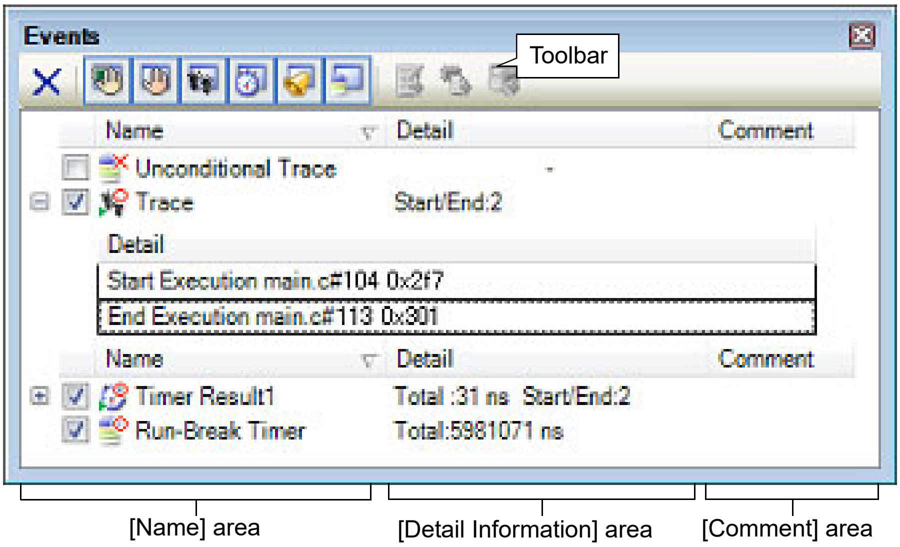
This section describes how to manage those events.

Select the [View] menu >> [Event].

Events are all managed in the Events panel. In the Events panel, you can confirm the details of the currently set events in a list, and they can be deleted and changed enable/disable status.

For details on the contents and function in each area, see the section for the Events panel.

Figure 2.93 Manage Events (Events Panel)



Remark 1. For "event occurrence" for a microcontroller that supports multi-core, see also to "2.9 Select a Core (PE)".

Remark 2. For details on how to set various events, see the section below:

- "2.11.3 Stop the program at the arbitrary position (breakpoint)"
- "2.11.4 Stop the program at the arbitrary position (break event)"
- "2.11.5 Stop the program with the access to variables/I/O registers"
- "2.14.3 Collect execution history in a section"
- "2.14.4 Collect execution history only when the condition is met"
- "2.15.2 Measure execution time in a section"
- "2.16.1 Measure the performance in a section"
- "2.18.1 Inset printf"

2.19.1 Change the state of set events (valid/invalid)

By changing the check on the check box of the event name, the setting state of the event can be changed (the Event mark is changed depending on the setting state of the event).

The following are types of the setting state of the event.

Figure 2.94 Event Name Check Box

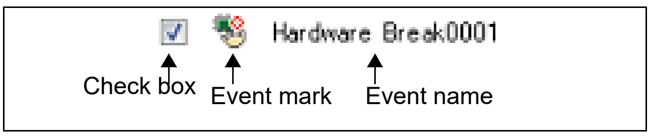


Table 2.22 Setting State of Event

	Valid state	Event occurs when the specified condition is met. It is possible to set the event to an invalid state by removing the check.
	Invalid state	Event does not occur when the specified condition is met. It is possible to set the event to a valid state by removing the check.
	Suspended State	A specified condition cannot be set in the program to be debugged. It is not possible to operate the check box.

Remark 1. Both of the timer start event and the timer end event is must be set for the Timer Result event. [Simulator]

Remark 2. It is not possible to set the Run-Break Timer event to an invalid/suspended state.

Remark 3. The setting state of the event can be changed from the menu displayed by right clicking on the Event mark in the Editor panel/Disassemble panel.

Remark 4. The setting of the Unconditional Trace event and the Trace event to valid or invalid state is exclusively controlled. Therefore, the Unconditional Trace event, which is a built-in event, is valid state by default, but if either a trace start event/trace end event is set, it automatically becomes invalid state, and the Trace event, which is a event name that is collectively called with a trace start event and a trace end event, becomes valid state. Conversely, if the set Trace event is invalid state, the Unconditional Trace event automatically becomes valid state.

2.19.2 Display only particular event types

Click on the toolbar button to display only the particular event type.

	Displays events related to the Hardware Break.
[Full-spec emulator][E1][E20]	Displays events related to the Software Break.
	Displays events related to the trace.
	Displays events related to the timer.
[Full-spec emulator][E1][E20]	Displays events related to the performance.
	Displays events related to the action event (Printf event).
	Displays the built-in events (Unconditional Trace event and Run-Break Timer event).

2.19.3 Jump to the event address

Clicking the following buttons jumps to each panel which selected events address exist.
Note however, that when a Trace event/Timer Result event/Performance Measurement event/Unconditional Trace event/Run-Break Timer event is selected, these buttons are disabled.

	Opens the Editor panel and jumps to the source line corresponding to the address where the selected event is being set.
	Opens the Disassemble panel and jumps to the disassemble results corresponding to the address where the selected event is being set.
	Opens the Memory panel and jumps to the source line corresponding to the address where the selected event is being set.

2.19.4 Edit detailed settings of events

This section describes how to edit detailed settings of various events.

2.19.4.1 Edit execution-related events

Remark For information on editing of action events (Printf events), see "2.18 Set an Action into Programs".

2.19.4.1 Edit execution-related events

The address condition and pass count condition **[Simulator]** for execution-related events you have set can be edited.

Perform this operation in the Detail dialog box (for execution events) that is opened by selecting [Edit Condition...] from the context menu after moving the caret to an execution-related event[Note] you want to edit on the Events panel.

Note An execution-related event refers to any one of the following events in the Events panel.

- Hardware Break event (execution type)
- Execution-related event as start and end condition in detailed information on Trace event
- Execution-related event as start and end condition in detailed information on Timer Result event
- Execution-related event as start and end condition in detailed information on Performance Measurement event

For the execution-related events that were set according to the descriptions in 2.11.3.1 Set a breakpoint, 2.11.4.1 Set a break event (execution type), 2.14.3.1 Set a Trace event, 2.15.2.1 Set a Timer Result event, and 2.16.1.1 Set a Performance Measurement event the initial values of the address and pass count conditions are listed below.

Table 2.23 Initial Values of Address Conditions

Item	Initial Value	
	Hardware Break	Trace, Timer Result, Performance Measurement
Compare condition	Before execution: Specified address (==) After execution: Inside the range (<=Addresses<=)	Inside the range (<=Addresses<=)
Address	*Address at the time of setting the event*	
Start address	*Address at the time of setting the event*	
End address	*Address at the time of setting the event*	
Use the address mask	No	
Mask value	0xFFFFFFFF	

Table 2.24 Initial Values of Pass count conditions

Item	Initial Value
Pass count [Simulator]	1

2.19.4.2 Edit access-related events

The address condition, data condition and pass count condition **[Simulator]** for access-related events you have set can be edited.

Perform this operation in the Detail dialog box (for access events) that is opened by selecting [Edit Condition...] from the context menu after moving the caret to an access-related event[Note] you want to edit on the Events panel.

Note An access-related event refers to any one of the following events in the Events panel.

- Hardware Break event (access type)
- Access-related event as start and end condition in detailed information on Trace event
- Access-related event in detailed information on Point Trace event

- Access-related event as start and end condition in detailed information on Performance Measurement event

For the access-related events that were set according to the descriptions in 2.11.5.1 Set a break event (access type), 2.14.3.1 Set a Trace event, 2.14.4.1 Set a Point Trace event, and 2.16.1.1 Set a Performance Measurement event the initial values of the address, data, and pass count conditions are listed below.

Table 2.25 Initial Values of Address Conditions

Item	Initial Value	
	Hardware Break	Trace, Point Trace, Performance Measurement
Compare condition	[Full-spec emulator][E1][E20] RH850G3M, RH850G3K, RH850G3MH, RH850G3KH: Specified address (==) RH850G4MH, RH850G4KH: When a value for comparison has been specified: Inside the range (<=Addresses<=) When a value for comparison has not been specified: Specified address (==) GTM: Inside the range (<=Addresses<=) [Simulator] Specified address (==)	[Full-spec emulator][E1][E20] Inside the range (<=Addresses<=) [Simulator] Specified address (==)
Address	*Address at the time of setting the event*	
Start address	*Address at the time of setting the event*	
End address	*Address at the time of setting the event*	
Use the address mask	No	
Mask value	0xFFFFFFFF	

Table 2.26 Initial Values of Data Conditions

Item	Initial Value
Access type	Access type at the time of setting the event
Access size	Variable or I/O register size at the time of setting the event
Compare condition	When a value for comparison has been specified : Specified value (==) When a value for comparison has not been specified : No conditions
Data	*Data at the time of setting the event*
Lower data	*Data at the time of setting the event*
Upper data	*Data at the time of setting the event*
Use the data mask	No
Mask value	0xFFFFFFFF

Table 2.27 Initial Values of Pass count Conditions

Item	Initial Value
Pass count [Simulator]	1

2.19.5 Delete events

To delete any event and event condition you have set, select the event and click the  button on the toolbar.

Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).

Remark 1. For the Break event of execution type, it is possible to delete the set event to click the event mark displayed in the Editor panel/Disassemble panel.

Remark 2. To delete all of the events and event conditions you have set at a time, select [Select All] from the context menu, then click the button (note, however, that it is not possible to delete the built-in events).

2.19.6 Write comment to events

The user can write comments for each event that has been set.
To input comments, click the [Comment] area after selecting the event to input comments, then input directly the desired text from the keyboard (the edit mode is cancelled by pressing the [Esc] key).
After editing the comments, complete the editing by pressing the [Enter] key or moving the focus to outside the edit region.
Up to 256 characters can be inputted for the comments, and this is saved as the settings of the user during use.

2.19.7 Notes for setting events

This section describes notes for setting each type of event.

2.19.7.1 Restrictions on the numbers of valid events and channels

The number of events that can be set to Valid state at the same time are subject to the following limitations.
Consequently, if enabling a new event would exceed the limit, you must first set some other event to Invalid state.

Table 2.28 Maximum Number of Enabled Events

Event Type	Debug Tool			
	Full-spec emulator/E1/E20			Simulator[Note 1]
	RH850G3M, RH850G3K, RH850G3MH, RH850G3KH	RH850G4MH, RH850G4KH	GTM	
Hardware Break (after execution)	8[Note 2] + 8[Note 3]	8[Note 2] + 8[Note 3]	2[Note 8]	64
Hardware Break (before execution)	12+4[Note 4, Note 5]	12+4[Note 4, Note 5, Note 6]	-	
Software Break	2000	2000	-	
Trace (trace start/trace end) Timer Result (timer start/timer end) Performance Measurement (performance measurement start/ performance measurement end)	8[Note 2] + 8[Note 3]	8[Note 2] + 8[Note 3]	-	
Point Trace	8[Note 3]	8[Note 3]	2[Note 8]	
Action (Printf)	100[Note 7]	100[Note 7]	-	

"*x* + *y*": "Execution type: *x*" + "Access type: *y*"

Remark 1. When a multi-core microcontroller is selected, the number of events are specified as follows.

- [Full-spec emulator][E1][E20]
  Events can be set per core up to the upper limit on the number of events.
  Software breaks can be set until the total number specified for all cores reaches the upper limit.
- [Simulator]
  Events can be set until the total number specified for all cores reaches the upper limit.

Remark 2. [Full-spec emulator][E1][E20]
The following restrictions apply to the number of access-type events for hardware breaks (after execution), access-type events for hardware breaks (before execution), tracing, timer measurement, performance measurement, and point tracing.

- When a range of addresses is specified or the unit of access is specified as 8 bytes, two events are available.
- When an unconditional trace event is enabled and data-access trace information is acquired, one access-related event is no longer available for use as other types of event since it is occupied in the acquisition of data.

Note 1. Hardware breaks (after execution), software breaks, and performance measurement cannot be set.

Note 2. These are execution-related events for hardware breaks (after execution), and can also be used as execution-related events for tracing, timer measurement, and performance measurement.

Note 3. These are access-related events for hardware breaks (after execution), and can also be used as access-related events for tracing, timer measurement, performance measurement, and point tracing.

Note 4. Four of the execution-related events for hardware breaks (before execution) can also be used as access-related events for hardware breaks (before execution).

Note 5. A post-execution break only occurs in the following cases.

- When the data condition is specified after selecting [Break Settings] >> [Set Read Break to] / [Set R/W Break to] from the context menu
- When a write access of the read-modify-write instruction is detected, after selecting [Break Settings] >> [Set Write Break to] / [Set R/W Break to] from the context menu

Note 6. Data cannot be specified.

Note 7. Combination with Software Break events.

Note 8. These are access-related events for hardware breaks (after execution), and can also be used as point tracing.

The following restrictions apply to the number of timer and performance measurement channels that can be set at the same time.

Table 2.29 Maximum Number of Enabled Channels

Event Type	Debug Tool			
	Full-spec emulator/E1/E20			Simulator
	RH850G3M, RH850G3K, RH850G3MH, RH850G3KH	until RH850G4MH2.0	after RH850G4MH2.1, RH850G4KH	
Timer Result (timer start/timer end)	3 (for each core)	3 (for each core)	7 (common for all cores)	8 (common for all cores)
Performance Measurement (performance measurement start/ performance measurement end)	4 (for each core)	4 (for each core)	8 (common for all cores)	-

2.19.7.2 Event types that can be set and deleted during execution

The following types of events can be set or deleted during program execution, or during tracer/timer execution.

Table 2.30 Event Types That Can be Set and Deleted during Execution

Event Type	Debug Tool		
	Full-spec emulator/E1/E20	GTM	Simulator
Hardware Break (after execution)	△	NG	-
Hardware Break (before execution)	△	-	▲
Software Break	NG	-	-
Trace (trace start/trace end) Point Trace	□	NG	▲
Timer Result (timer start/timer end)	NG	-	▲
Performance Measurement (performance measurement start/ performance measurement end)	NG	-	-
Action (Printf)	NG	-	▲

△ : Possible, if the program execution is allowed to pause for events[Note]
▲ : Impossible while tracer or timer is executing
□ : Impossible while tracer or timer is executing. Possible, if the program execution is allowed to pause for events[Note]
NG : Impossible
– : Not supported

Note To enable this, specify [Yes] with the [Set event by stopping execution momentarily] property in the [Set Event While Running] category on the [Debug Tool Settings] tab of the Property panel.

2.19.7.3 Other notes

- No events can be set to local variables.

- Events do not occur during step execution (including return execution) and program execution by selecting [Go to Here] from the context menu.

- If the location set for an existing event changes to midway in an instruction because the program to debug has been downloaded again, re-set the event using the following method:
  - When debugging information is available:
    The location setting of events is always moved to the beginning of the source text line.
  - When debugging information is not available:
    Depends on the [Automatic change method of event setting position] property in the [Download] category on the [Download File Settings] tab of the Property panel.
- If a change to internal ROM/RAM changes the location the event is set to a non-mapped area, then set events will not occur (they will also not change to Invalid state /Suspended State on the Events panel).

- If you differentiate function or variable names by leading underscores, then CS+ may misrecognize them, and convert symbols or make break event settings invalid. This applies for cases like when you have two functions, one named "_reset" and the other named "__reset".

2.20 Use Hook Function

This section describes how to set hooks in the debug tool by using the hook function.

By setting a hook transaction, you can automatically change the values of the I/O register/CPU register before and after downloading a load module or after resetting the CPU.

Configure the hook transaction in the [Hook Transaction Settings] category on the [Hook Transaction Settings] tab of the Property panel.

Remark By setting a I/O register by using the [Before download] property, for example, downloading can be executed at high speeds. Downloading to the external RAM is also facilitated by using this function.

Figure 2.95 [Hook Transaction Settings] Category

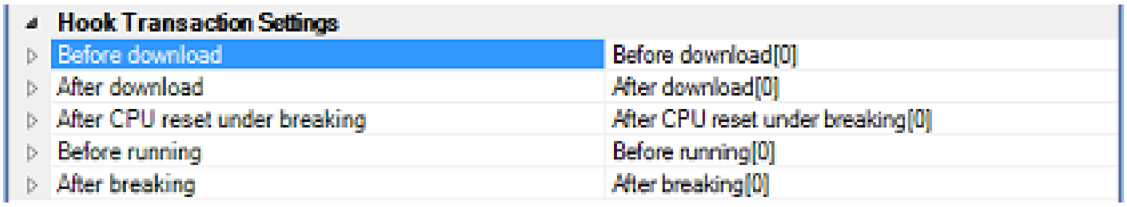


Table 2.31 Properties in [Hook Transaction Settings] Category

Property	Description
Before download	Perform the specified process immediately before downloading the load module file.
After download	Perform the specified process immediately after downloading the load module file.
After CPU reset under breaking	Perform the specified process immediately after resetting the CPU under breaking.
Before running	Perform the specified process immediately before starting program execution.
After breaking	Perform the specified process immediately after breaking program execution.

The properties in the [Hook Transaction Settings] category indicate the timing with which the hook process will be performed. "[]" indicates the current number of specified processes (no hook processes are configured by default).

Specify the target process in the property for which you want the hook process to be performed.

To specify a process, select the target property, then open the Text Edit dialog box by clicking the [...] button that appears on the right edge of the field.

Figure 2.96 Opening Text Edit Dialog Box

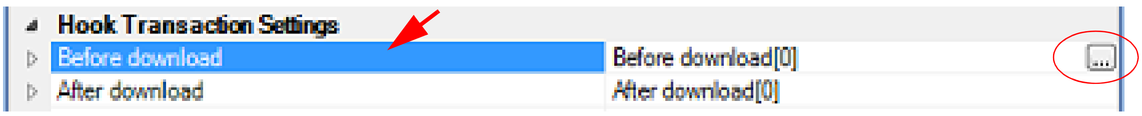
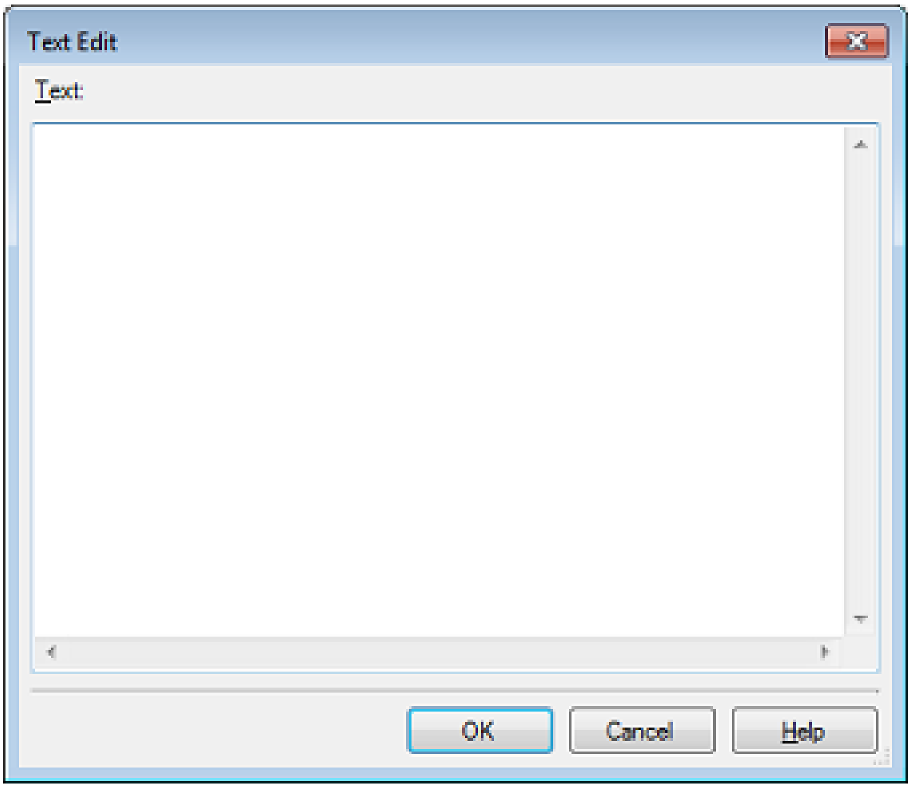


Figure 2.97 Use Hook Function (Text Edit Dialog Box)



In this dialog box, directly enter the desired process from the key board.
The format for specifying processes is as follows:

[Process 1]:
Automatically overwrites the value of *I/O register* with *Value*.
Specification format:

```
I/O-register-name Value
```

[Process 2]:
Automatically overwrites the value of *CPU register* with *Value*.
Specification format:

```
CPU-register-name Value
```

[Process 3]:
Automatically executes a script file which is specified with *Python script path* (absolute path or relative path from the project folder).
Specification format:

```
Source Python-script-path
```

Remark 1. When specifying hook processes, lines starting with a hash mark "#" will be treated as comments.

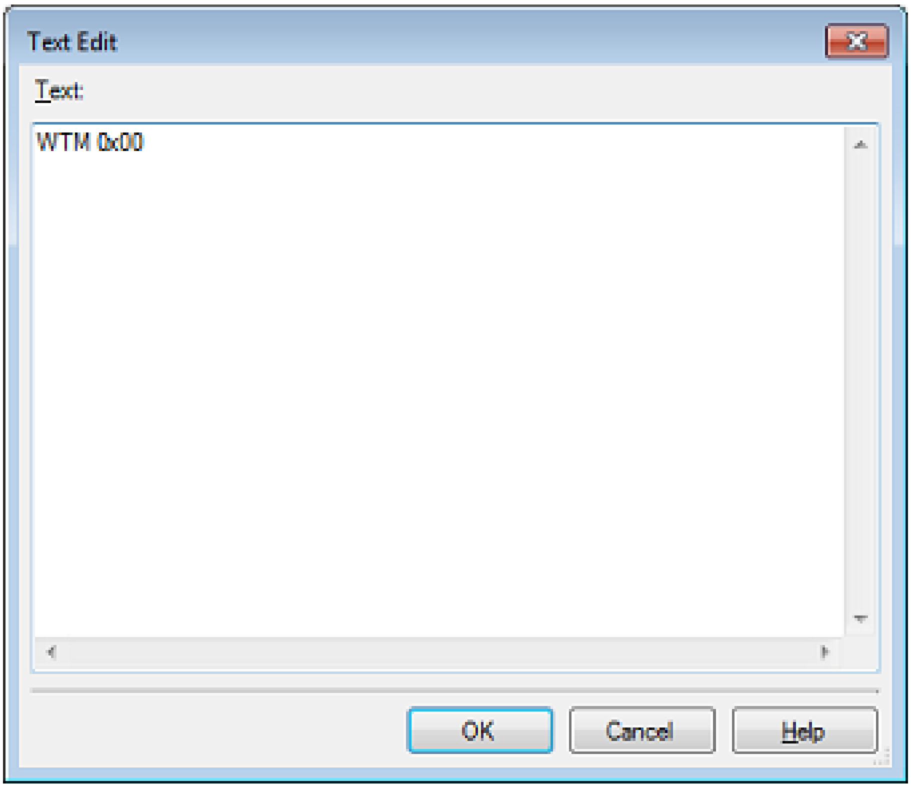
Remark 2. A tab character can be used instead of the space character.

Caution You can use the following commands when you execute the Python script in the Hook process of the debugger.
debugger.Register.GetValue
debugger.Register.SetValue
debugger.Memory.Read
debugger.Memory.Write
If you want to use other Python command, please use Hook command in the Python console.

Up to 64 characters for one process, and up to 128 processes for each property can be set (one line in the [Text] area in the Text Edit dialog box is equivalent to one processing).

After the specification of the process is complete, click the [OK] button to set the process to the Property panel.

Figure 2.98 Example of Hook Transaction



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/dialog box.

(1) Character set
Character sets that are allowed to input are as follows:

Table 2.32 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 Big-Endian	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji (include Chinese characters) and 1-byte Katakana.
UTF-16 Little-Endian	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji (include Chinese characters) and 1-byte Katakana.
UTF-32 Big-Endian	2-byte alphabet, number, symbol, Hiragana, Katakana, Kanji (include Chinese characters) and 1-byte Katakana.
UTF-32 Little-Endian	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 2.33 Escape Sequence List

Escape Sequence	Value	Outline
\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.

Escape Sequence	Value	Outline
\\	0x5C	Backslash

(3) Number
Notations allowed when entering numbers are as follows:

Table 2.34 Notation List

Notation	Outline
Binary number	Start with 0b and continues with the numbers from 0 to 1. (Case insensitive for alphabets)
Octal number	Start with 0 and continues with the numbers from 0 to 7.
Decimal	Start without 0 and continues with the numbers from 0 to 9.
Hexadecimal number	Start with 0x and continues with the numbers from 0 to 9 and alphabets a to f. (Case insensitive for alphabets) In the input area with the HEX mark, prefix 0x is not needed.

(4) Expression and operator
Expression represents constants, CPU register name, I/O register name and symbols and those connected with operators.
An expression comes in two types: an address expression and a watch-expression. The expression that requires the address of a symbol is referred to as an address expression, and the one that requires the value of a symbol is referred to as a watch-expression.

(a) An address expression and operators
With an address expression, the address of a symbol is used to perform operations. Only when a CPU register name is written, the value of the symbol is used to perform operations.
The basic input formats of address expressions are as follows:

Table 2.35 Basic Input Format of Address Expressions

Expression	Description
Name of a C language variable[Note 1]	Address of a C language variable
Expression [*Expression*[Note 2]]	Address of an array
Expression.Member name	Address of a structure/union/class member
Expression->Member name	Address of a structure/union/class member that is pointed to
Name of a CPU register	Value of the CPU register
Name of an I/O register	Address of the I/O register
Label name[Note 3], EQU symbol name[Note 3] and [immediate value]	Address of a label, a value of an EQU symbol, and an immediate address
Integer constant	Address

Note 1. If the register is assigned the value of a C variable, an error results.

Note 2. The expression that is input as an index to an array is parsed as a watch-expression.

Note 3. If the label name or EQU symbol name includes a "$", be sure to enclose the name in "{ }" (e.g. {$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 2.35 Basic Input Format of Address Expressions", the following expressions with operator can be constructed.

Table 2.36 Construction of Expressions with Operators

Expression	Description
(*Expression*)	Value of the parenthetical watch-expression
! *Expression*	Inverts symbol
- *Expression*	Logical negation
~ *Expression*	Bit inversion
Expression * *Expression*[Note]	Multiplication
Expression / *Expression*[Note]	Division
Expression % *Expression*[Note]	Remainder calculation
Expression + *Expression*[Note]	Addition
Expression . *Expression*[Note]	Subtraction
Expression & *Expression*[Note]	Logical multiplication by bits
Expression ^ *Expression*[Note]	Exclusive disjunction by bits
Expression | *Expression*[Note]	Logical sum by bits

Note Variables and functions can be combined by an operator only with variables, functions and integer constants.

Example C variable name + 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 (e.g. main( ) + 1).
The basic input formats of watch-expressions are as follows:

Table 2.37 Basic Input Format of Watch-expressions

Expression	Description
Name of a C language variable	Address of a C language variable
Expression [*Expression*	Element of an array
Expression.Member name	Value of a structure/union/class member
Expression->Member name	Value of a structure/union/class member that is pointed to
**Expression*	Value of pointer variable
&*Expression*	Location address
Name of a CPU register	Value of the CPU register
Name of an I/O register	Value of the I/O register
Label name[Note], EQU symbol name[Note] and and [immediate value]	Values of a label, a value of an EQU symbol, and an immediate address
Integer constant	Integer constant value
Floating constant	Floating point constant value
Character constant	Character constant value

Note If the label name or EQU symbol name includes a "$", be sure to enclose the name in "{ }" (e.g. {$Label}).
Any imaginary number must be multiplied by an uppercase "I" (e.g. 1.0 + 2.0*I). When you spec-

ify 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 2.37 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 2.38 Construction of Expressions with Operators

Expression	Description
(*Expression*)	Specifies the order in which operations are performed
! *Expression*	Inverts symbol
- *Expression*	Logical negation
Expression * *Expression*Note	Multiplication
Expression / *Expression*Note	Division
Expression % *Expression*Note	Remainder calculation
Expression + *Expression*Note	Addition
Expression . *Expression*Note	Subtraction
Expression & *Expression*Note	Logical multiplication by bits
Expression ^ *Expression*Note	Exclusive disjunction by bits
Expression | *Expression*Note	Logical sum by bits

Note Variables and functions can be combined by an operator only with variables, functions and integer constants.
C variable name + I/O register name

Caution 1. 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 2. 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 3. 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 4. 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.

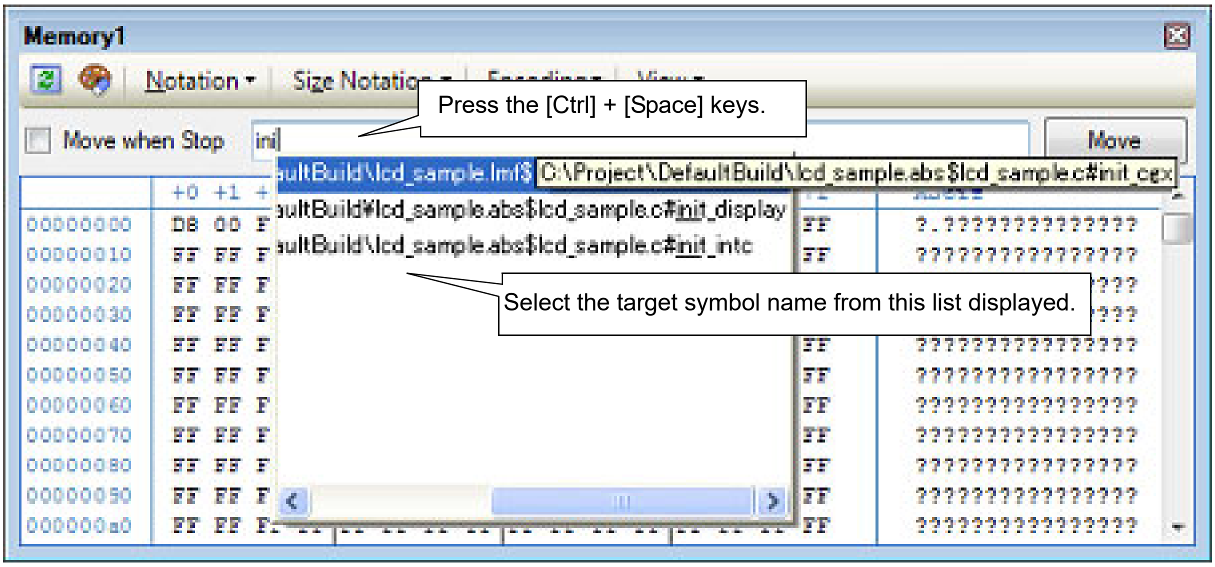
At this time, if a key other than the [Space]/[Enter] key is pressed or the focus moves to outside the panel/dialog box currently being operated, then the list of symbol names will disappear (the symbol name completion will not be performed).

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. Since the information for use by the symbol name completion function is generated while symbols are being downloaded, the time taken for downloading and the memory usage on the host machine will increase when this function is enabled. Therefore, if you do not intend to use the symbol name completion function, we recommend invalidating this function by selecting [No] in the [Generate the information for input completion] item in the Download Files dialog box ([Yes] is selected by default).
Note, however, that if GHS compiler is used, it is not possible to invalidate this function (a specification of the [Generate the information for input completion] item will be ignored).

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 2.99 Symbol Name Completion Function



2.21.3 Icons for invalid input

In some of the dialog boxes in 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 Saving and Restoring the States of Debug Tools

Saving the states of debug tools (readable and writable memory and register values) enables restoring the states in which the state information for the debug tools was saved.

Remark If you have selected [Save the value of SFRs and IORs added in the Watch 4 panel] in the [General - Debug] category of the Option dialog box, the values of SFRs and IORs will also be saved.

Saving and restoring the states of debug tools is mainly used to realize the following two features.

- Rewinding the state of the debug tool to restore the state when the execution of a program was started after the program has been repeatedly run and stopped
- Restoring the state to any among multiple states that have been stored

(1) Rewinding the state of the debug tool
The state of the debug tool can be restored to that before the program started running after the program has been repeatedly run and stopped.

[Usage]

(a) Select the [Enable rewind feature] checkbox in the [General - Debug] category of the Option dialog box.

(b) The state of the debug tool is automatically saved when the program starts running.

(c) After the program has been repeatedly run and stopped, select [Rewind debug tool state] from the [Debug] menu or press the [Ctrl] + [F11] keys to restore the state to that before the program started running.

Remark A maximum number of times of rewinding can be specified with [Maximum number of times of rewinding] in the Option dialog box.

(2) Saving and restoring the states of multiple debug tools
The states of multiple debug tools can be saved and restored.
For example, you can save the states for functions being debugged which are not frequently called and restore the states for debugging many times.

[Usage]

(a) Save the states of debug tools in one of the following three ways.

- Select [Save debug tool state *1-4*] from the [Save debug tool state] menu under the [Debug] menu.
- Select [Save debug tool state *1-4*] on the [State save event] tab in the Action Events dialog box. Executing the instruction at the address specified in the dialog box saves the states.
- Run debugger.DebugTool.SaveState(*fileName*) in the Python console to save the states of debug tools in a file.

(b) Restore the states of debug tools in one of the following three ways.

- Select [Restore debug tool state *1-4*] from the [Save debug tool state] menu under the [Debug] menu.
- Select [Restore debug tool state *1-4*] on the [State save event] tab in the Action Events dialog box. Executing the instruction at the address specified in the dialog box restores the states and continues the execution of the program from the saved states.
- Run debugger.DebugTool.RestoreState(*fileName*) in the Python console to restore the states of debug tools from a file.

2.23 Exclusive Control Checking Tool

The exclusive control checking tool checks whether there is any function that accesses global variables (except static variables) outside the exclusive control period. In other words, it is a tool used to check whether any function is directly accessing global variables (except static variables) without using the exclusive control mechanism.

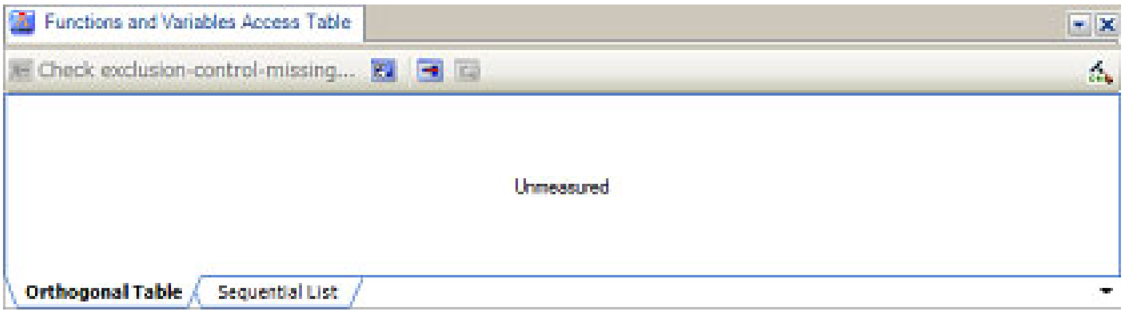
Caution 1. This tool can be supported when CC-RH V1.04.00 or later is in use.

Caution 2. This function does not check whether there is a problem with the exclusive control mechanism (mechanism to prohibit variables from being accessed by other functions during the exclusive control period).

[How to use]

(1) Open the Functions and Variables Access Table panel
Open the Solution List panel and then click the [GO] button of Exclusive control checking tool. The Functions and Variables Access Table panel will open.

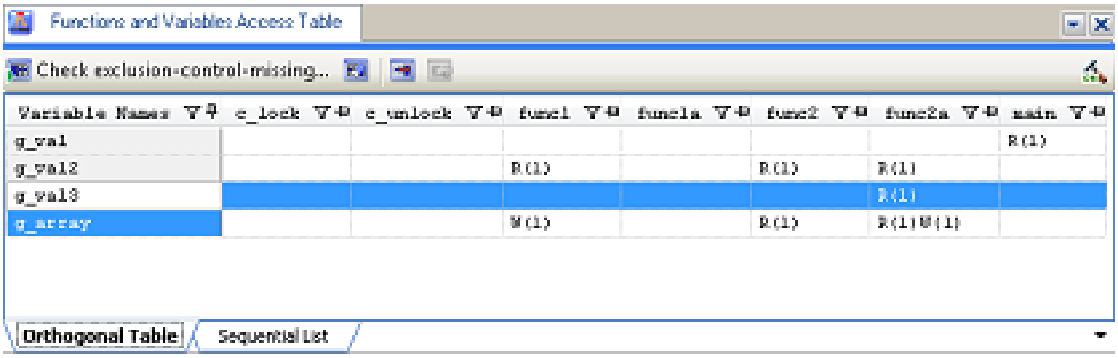
Figure 2.100 Functions and Variables Access Table Panel



(2) Make preparations for checking of exclusive control
The preparations that have to be made before checking of exclusive control starts are explained below.

(a) Run a build with the cross reference information set to be output
Click on the toolbar on the Functions and Variables Access Table panel to configure cross reference information on which variables were accessed by the functions.
When building completes successfully, an orthogonal table of the functions accessing global variables (except static variables) is generated.

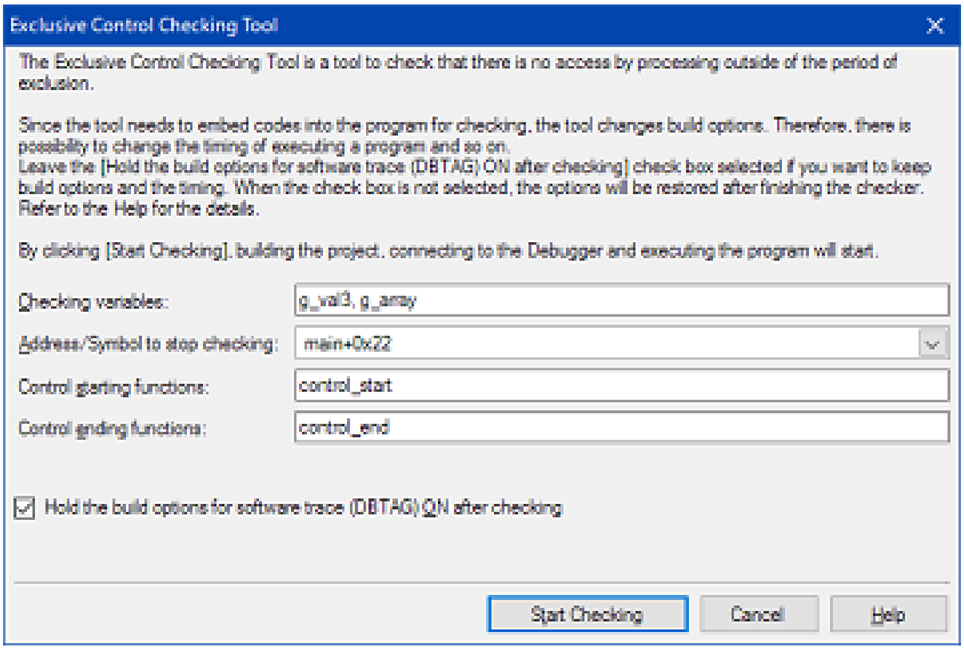
Figure 2.101 Functions and Variables Access Table Panel



(b) Select the variables to be checked
Select the variables to be checked from the orthogonal table. More than one variable can be selected.
"R" is displayed if the function has read a value from the variable and "W" is displayed if the function has written a value to the variable. The number in parentheses indicates from how many locations the variable was accessed.

(c) Open the Exclusive Control Checking Tool dialog box
On the Functions and Variables Access Table panel, click the [Check exclusion-control-missing...] button on the toolbar. The Exclusive Control Checking Tool dialog box will open.

Figure 2.102 Exclusive Control Checking Tool Dialog Box



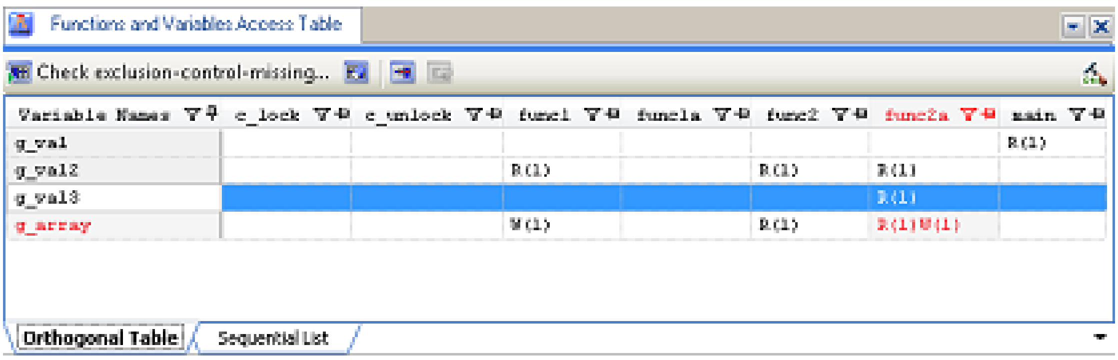
(d) Set an address or symbol where checking will end
Checking is carried out by executing the user program and recording and analyzing information on accesses to variables. Therefore, where to stop checking needs to be specified. An address or symbol can be set.

(e) Set the functions for controlling accesses to variables
Set the functions (control starting function and control ending function) for controlling accesses to variables. The control starting function is used to disable accesses to variables and the control ending function is used to enable accesses to variables.

(3) Start checking
Click the [Start Checking] button.
Information on accesses to variables is obtained by embedding a software trace instruction in the program and analyzing how the embedded software trace instruction was executed. Therefore, information on accesses to variables that was acquired with a software trace instruction embedded in the program sometimes does not match access information that was acquired without a software trace instruction embedded in the program, in relation to the timing of program execution. If you wish to make access information match, leave [Hold the build options for software trace (DBTAG) ON after checking] selected.

(4) Confirm the checking result
After checking has completed, locations where exclusive control did not work have been detected are shown in error color on the Functions and Variables Access Table panel. The relevant location can be opened in an editor by double-clicking or pressing the [Enter] key at that location and the erroneous code can be found immediately.

Figure 2.103 Functions and Variables Access Table Panel



Caution Note the following points when using the build options for software trace (DBTAG).

- The -Xcref option of the compiler and the -list -show option of the linker are automatically added (does not affect the load module generated as a result of building) when the exclusive control checking tool is used.
- Since the specified variables are handled as if they were volatile-declared, the optimization result may differ.
- The DBTAG instruction to be embedded is equivalent to a NOP instruction. Though the memory and register values do not change, there is the slightest difference in the timing related to program execution.
- When the exclusive control checking tool is used, do not describe a DBTAG instruction in the user program.
  In such a case, the tool cannot check whether exclusive control is performed correctly.

2.24 Pseudo-error Debugging [Full-spec emulator][E1][E20]

Pseudo-error debugging is a feature for generating pseudo-errors which are difficult to reproduce on the real machine. You can use this feature to check the operation of the reset routine or handlers called in response to errors, or functions to be called from within handlers, and then debug the code.

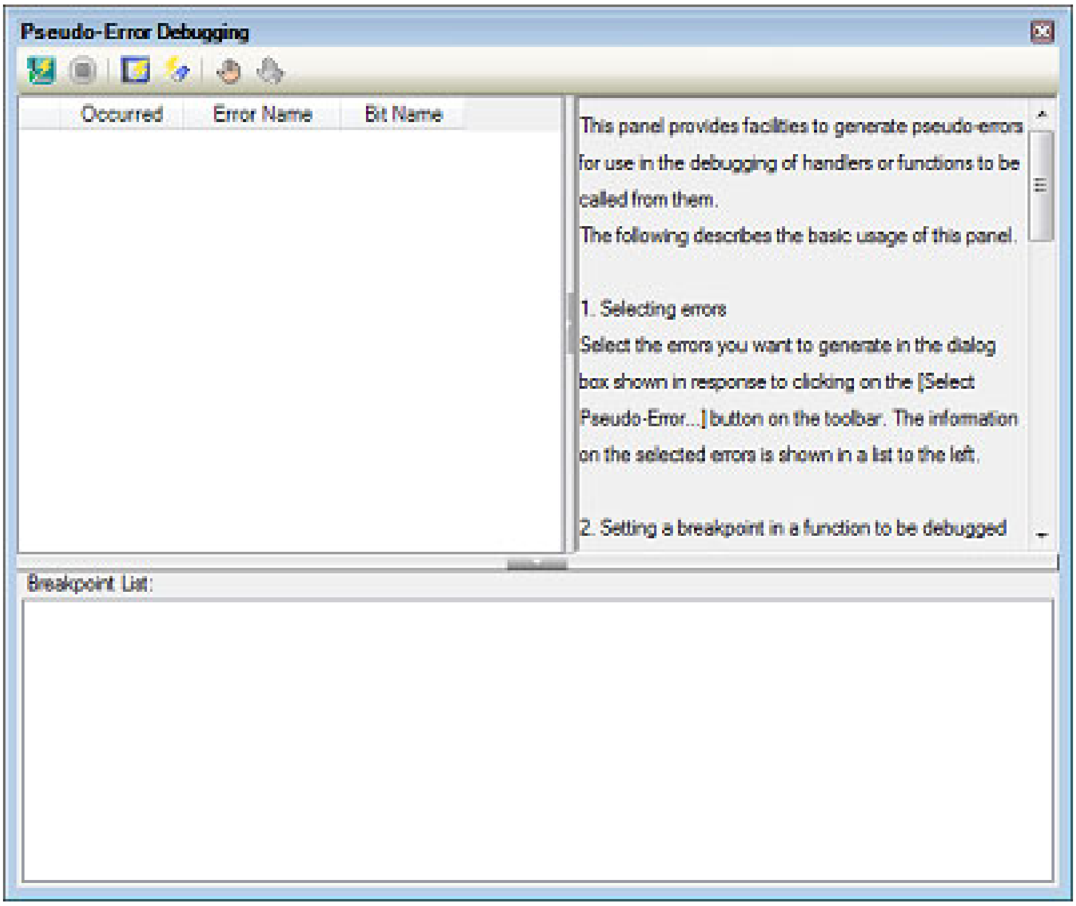
The Error Control Module (ECM)[Note] is used to generate such pseudo-errors.

Note See the chapter of the ECM in User's Manual: Hardware of each device for details. A device without the ECM does not have a chapter of the ECM in its manual. In such a case, pseudo-error debugging cannot be used.

[How to use]

(1) Open the Pseudo-Error Debugging panel
Open the Solution List panel and click the [Pseudo-Error Debugging] button to display the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].

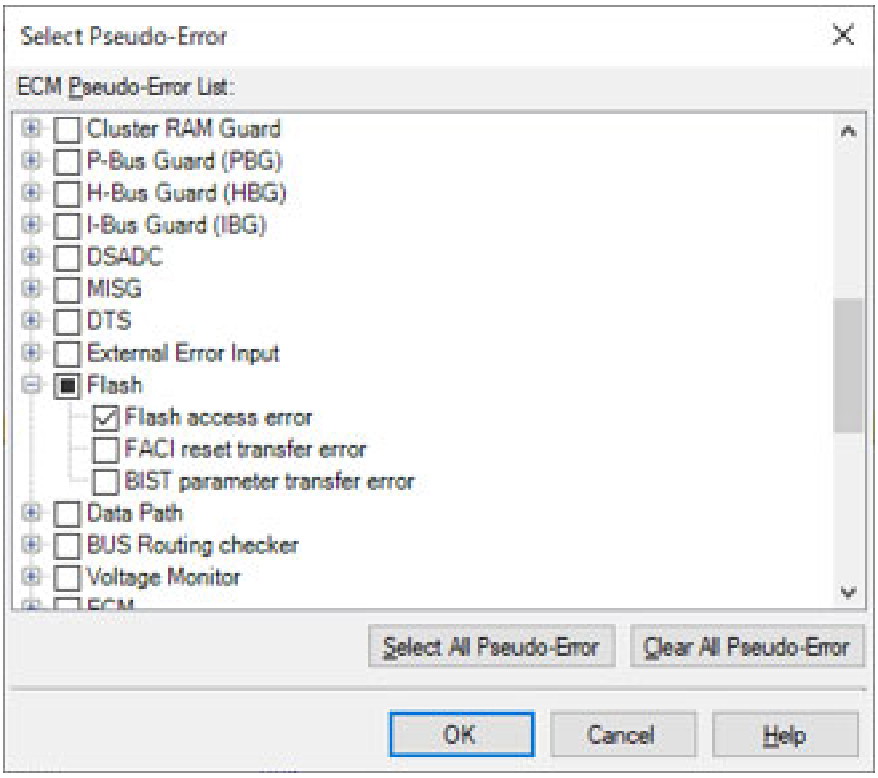
Figure 2.104 Pseudo-Error Debugging Panel



Remark For a device that does not support pseudo-error debugging, pseudo-error debugging is not displayed on the Solution List panel. Even for a device that supports pseudo-error debugging, the [Pseudo-error Debugging] button is nullified when the simulator is being used or the debug tool has not been connected.

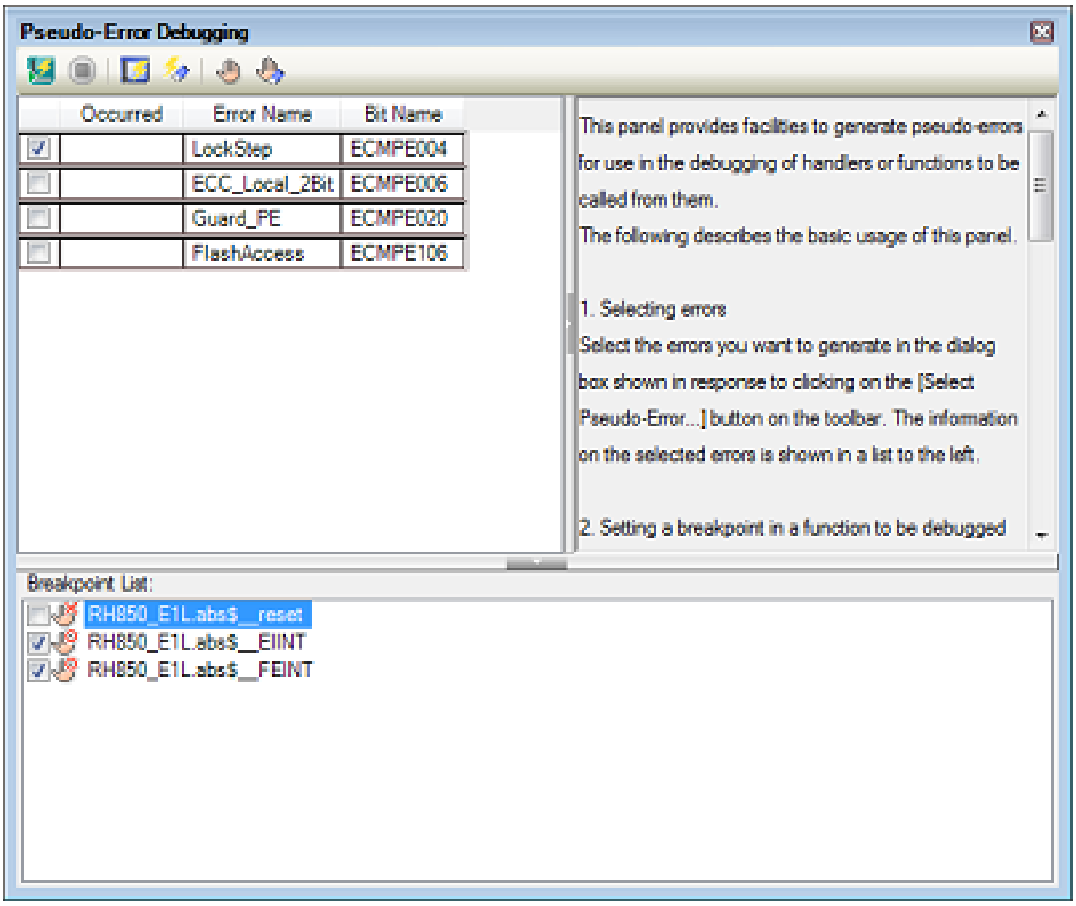
(2) Open the Select Pseudo-Error dialog box
On the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20], click . The Select Pseudo-Error dialog box [Full-spec emulator][E1][E20] will open.

Figure 2.105 Select Pseudo-Error Dialog Box



(3) Select the error you wish to generate
A list of errors supported by the pseudo-error debugging feature is displayed in the Select Pseudo-Error dialog box [Full-spec emulator][E1][E20]. Select the check box of the error you wish to generate and click the [OK] button.
The selected error is displayed in the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].

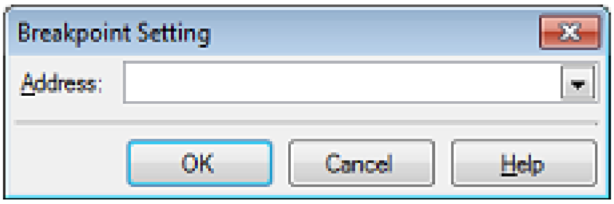
Figure 2.106 Pseudo-Error Debugging Panel



Remark See User's Manual: Hardware for details on errors. [Bit Name] shows the names of individual bits in ECM pseudo error trigger registers (e.g. ECMPE0) listed in the chapter of the ECM.

(4) Set the address where you wish to generate a break
Click on the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20] to open the Breakpoint Setting dialog box [Full-spec emulator][E1][E20]. Entering the address expression of a handler or reset routine to be called in response to an error enables a break to occur at that address after the error is generated. A symbol, such as a handler name, can also be entered.

Figure 2.107 Breakpoint Setting Dialog Box



Remark 1. Enter an address in the ROM area. A breakpoint cannot be set if an address in the RAM area is specified.

Remark 2. A software breakpoint is set at the specified address.

Remark 3. Clicking will not cause a break to occur at a breakpoint that was not set in this dialog box. The program can be forcibly stopped.

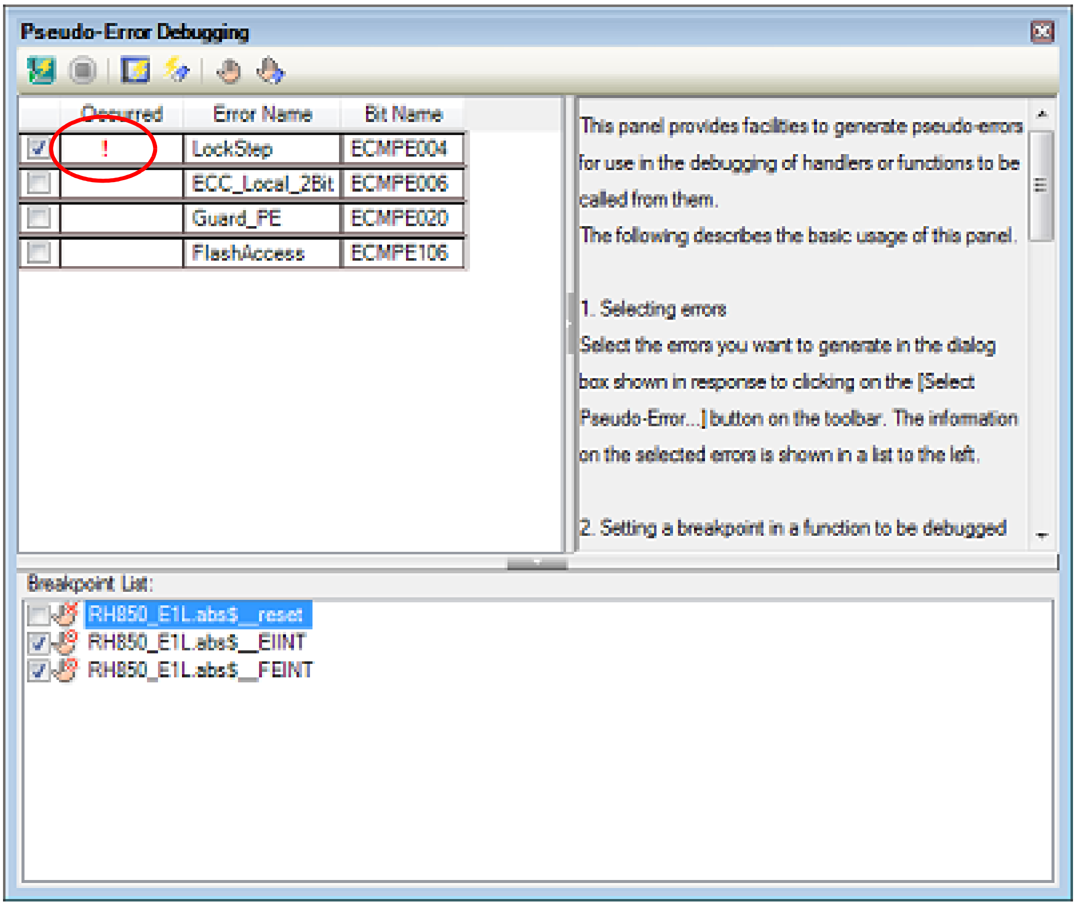
(5) Generate a pseudo-error
Click on the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20]. An error is generated using an ECM pseudo-error trigger register (e.g. ECMPE0). An interrupt or reset is generated based on the user setting. A break occurs when execution passes the address specified in the Breakpoint Setting dialog box [Full-spec emulator][E1][E20].

Remark Pseudo-error debugging is not supported in Async debug mode. When Async debug mode has been selected, click after switching the value of the [Debug mode] property in the [Multi-core]

category on the [Debug Tool Settings] tab of the Property panel to [Sync debug mode].
When there is an ECM pseudo error trigger register (e.g. ECMnPE0) for each PE, the ECM pseudo error trigger register (n in ECMnPE0 indicates the current core) of the current core is used.

(6) Display "!" for the generated error
When a break occurs, the ECM master/checker error source status register (e.g. ECMmESSTR0) is referenced and "!" is displayed for the generated error.

Figure 2.108 Pseudo-Error Debugging Panel



Remark 1. If the setting of the ECM master/checker error source status register (e.g. ECMmESSTR0) has been cleared, [!] is not displayed even when an ECM error occurs.

Remark 2. A bit is always assigned per error to indicate whether an error occurred. Two bits are assigned to some errors and the number of occurrences of an error that can be counted is up to three (e.g., this is the case for errors No. 0 to 7 in RH850/G4MH devices). If the pseudo-error has occurred several times, “!” will be displayed as the number of times the error has occurred. To check the number of bits that are assigned to an error, see “ECM Error Source Status Register” in the chapter of the ECM in User's Manual: Hardware for the MCU in use.

Caution 1. To cause an RH850/P1x-series MCU to be internally reset by a pseudo-error, set up an ECM pseudo-error trigger register from the user program. A special instruction sequence is required for writing to this register since it is protected. For details, refer to the section of write-protected registers in User’s Manual: Hardware for the MCU in use.

Caution 2. Do not set a breakpoint at the current PC address in pseudo-error debugging. When it is set, interrupts and internal reset will not occur.

Caution 3. If you wish to set another breakpoint via a source editor or the Disassemble panel at the address where a breakpoint has been set for pseudo-error debugging, designate it as a hardware breakpoint. Setting a software breakpoint in this case will lead to an error when pseudo-error debugging is started.

Caution 4. The following pseudo-error debugging wasn't being supported in RH850/D1L group and RH850/D1M group.

- FACI Reset transfer error (FRTERR)

- Flash sequencer error (FLERR)

2.25 Debugging CAN Bus Reception Procedures [Full-spec emulator][E1][E20]

Debugging of CAN bus reception is a feature for facilitating debugging of the reception procedure of a channel by continuously transmitting CAN bus frames to a desired channel by using the inter-channel communications facility of the RS-CAN.

Caution This feature can be used when the target device has RS-CAN units which consists two or more channels. This feature cannot be used when the device has RS-CAN FD units. (RS-CAN FD units behave as RS-CAN when the interface mode is classical CAN mode. But this feature is not supported because their units are different.)

[How to use]

(1) Open the Debugging CAN Bus Reception Procedures panel
Open the Solution List panel and click the [Debugging CAN Bus Reception Procedures] button to open the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].
Operation of the CAN bus reception procedure for an individual microcontroller can be verified using this feature.y

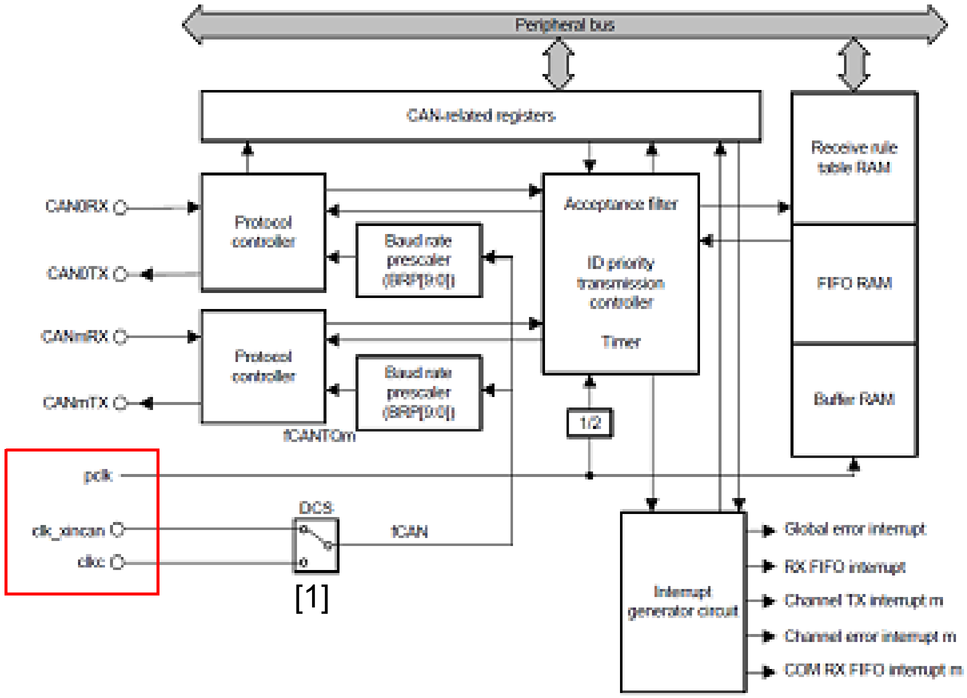
Remark For a device that does not support , Debugging CAN bus reception procedures is not displayed on the Solution List panel.
Even for a device that supports Debugging CAN bus reception procedures, the [Debugging CAN Bus Reception Procedures] button is nullified when the simulator is being used or the debug tool has not been connected.

(2) Make preparations for debugging of CAN bus reception
The preparations that have to be made before starting debugging of CAN bus reception are explained below.

(a) Set up the entire RS-CAN
Select the clock source and specify whether to use the timestamp, DLC checking function, and DLC replacement functions as the settings of the entire RS-CAN (see "RS-CAN Module Setting dialog box [Full-spec emulator][E1][E20]").
The meaning of each setting is as follows:

<1> Select the clock source
As shown in the figure below, the RS-CAN has three types of clock input: clk_xincan, clkc, and pclk.

Figure 2.109 RS-CAN Clock



Note: m = 0 to 3
BRP[9:0]: Bits in the RSCAN0CmCFG register
DCS: Bits in the RSCAN0GCFG register
fCANTQm: CANmTq clock
fCAN: CAN clock

For clk_xincan and clkc among the clock inputs, the clock that is to be input to the baud rate prescaler of each channel needs to be selected with switch [1] in the figure.
The clock source for the microcontroller to which each clock input is connected differs depending on the device. See the hardware manual of the device for details.

<2> Timestamp function
This function stores the timestamp of the frame reception time in a receive buffer or receive FIFO.
At debugging of CAN bus reception, the input clock of the timer for timestamp is a clock obtained by dividing the frequency of pclk by 2.

<3> DLC check function
Set whether to apply filtering by the Data Length Code (DLC) when receiving a frame in the RS-CAN.
Since this setting is effective for not each channel but the entire RS-CAN module, this has been made a setting of the entire RS-CAN.
When the DLC checking function is used, a frame larger than or equal to the data size set in the [Receive rule settings] area of the Receive Channel Setting dialog box [Full-spec emulator][E1][E20] can pass the filter.
Make this setting in accordance with your system.

<4> DLC replacement function
When the frame passes the filter of the DLC checking function, the DLC code is replaced with a value of the data size defined by receive rules.
This function is enabled only when the DLC checking function is used.

Remark When the data size of the frame that was received using the DLC replacement function is larger than or equal to the data size defined by receive rules, DLC is replaced with 0x0.

(b) Set up the receive channel
Set the channel number, reception speed, number of receive buffers to be used, receive FIFO number to be used, and receive rules of the receive channel that is the target of debugging of CAN bus reception in the Receive Channel Setting dialog box [Full-spec emulator][E1][E20].

The details are as follows:

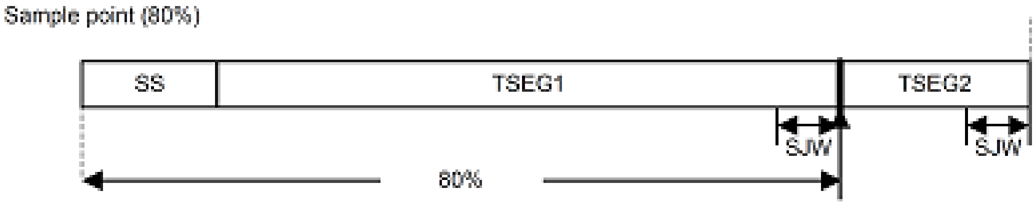
<1> Specify the receive channel number
Specify the number of the receive channel for which debugging of CAN bus reception is to be performed.
Specify the channel to call the reception procedure you wish to debug.

<2> Reception speed
The reception speed is determined by setting the frequency division ratio of the baud rate prescaler, the time unit of a propagation time segment, the time unit of a phase buffer segment, and the time unit of the resynchronization jump width.

The baud rate prescaler sets the frequency division ratio of the clock selected in "<1> Select the clock source".
The cycle of this divided clock is one time unit (Tq).

The relationship between the sample time for the RS-CAN to acquire one bit of data, the propagation time segment, the phase buffer segment, and the resynchronization jump width is shown below. The final reception speed is determined by this relationship.

Figure 2.110 RS-CAN_1 Bit Sample Time



SS	Synchronization segment The SS is a segment that performs synchronization by monitoring the edge from recessive to dominant bits in the Interframe Space. The value is fixed to 1Tq in the RS-CAN.
TSEG1	Propagation time segment TSEG1 is a segment that absorbs physical delay on the CAN network. Set a value within the range of 4 to 16 Tq.
TSEG2	Phase buffer segment TSEG2 is a segment that compensates phase error due to an error in frequency. Set a value within the range of 2 to 8 Tq. A value smaller than TSEG1 must be set.
SJW	Resynchronization jump width The SJW is a length to extend or reduce the time segment to compensate for an error in phase due to phase error. Set a value within the range of 1 to 4 Tq. A value smaller than or equal to TSEG2 must be set.

<3> Number of receive buffers to be used
In the RS-CAN, the receive buffers for each channel is a continuous area. When the number of receive buffers is specified, it means that the receive buffers with the number of 0 to (Receive buffer count - 1) are specified.
A receive buffer holds a single frame of data per one number.

<4> Receive FIFO number to be used
FIFOs can be used to store received frames in the RS-CAN. 128 frames of data can be stored in a single FIFO.

<5> Receive rules
These are filter settings for sorting received frames into receive buffers or FIFOs.
Filtering is based on the ID, frame type, and data size.
By specifying a label for each receive rule, you can identify the receive rule that was applied.

(c) Set up the transmit channel
Set the channel number used to transmit frames for debugging and the interval time for continuously transmitting frames in the Transmit Channel Setting dialog box [Full-spec emulator][E1][E20].
The details are as follows:

<1> Specify the transmit channel number
One channel is occupied for transmitting frames because the inter-channel communication facility of the RS-CAN module is used for debugging of CAN bus reception.
Specify the number of an unused channel or a channel that is not a target of debugging.

<2> Interval time of continuous transmission
The base clock of the interval time for continuous transmission is a clock obtained by dividing the frequency of pclk by 2, as shown in the figure in "<1> Select the clock source".
Specify the division ratio for the base frequency, how many clock cycles of the divided clock are to be used, or whether the interval time should be multiplied by 10.

(d) Set the transmit frame
Set the frame to be transmitted in the Transmit Frame Setting dialog box [Full-spec emulator][E1][E20].
The CAN bus frames that can be specified in debugging of CAN bus reception are only data frames and remote frames.
Set the type, ID, data size, and data of the frame you wish to transmit to the channel that is a target of debugging.
The set frame will be transmitted at the interval time set in "<2> Interval time of continuous transmission".

(e) Specify the timing to apply the settings
Specify the timing to apply the settings that have been set so far.
For the timing to apply the settings, specify immediately after debugging of CAN bus reception is executed or upon execution of the instruction at the specified address.
Since debugging of CAN bus reception changes the settings of the RS-CAN module, the settings should be applied after the RS-CAN module setting process in the user code has finished.

(3) Start debugging of CAN bus reception
Debugging of CAN bus reception can be started by setting a breakpoint at the address of a function in the reception procedure that is to be debugged and then clicking from the toolbar on the panel.

(4) Perform debugging after the program stops
Normal debugging can be performed after the program stops at the set breakpoint.

2.26 Measuring CAN Bus Reception Processing Times [E2]

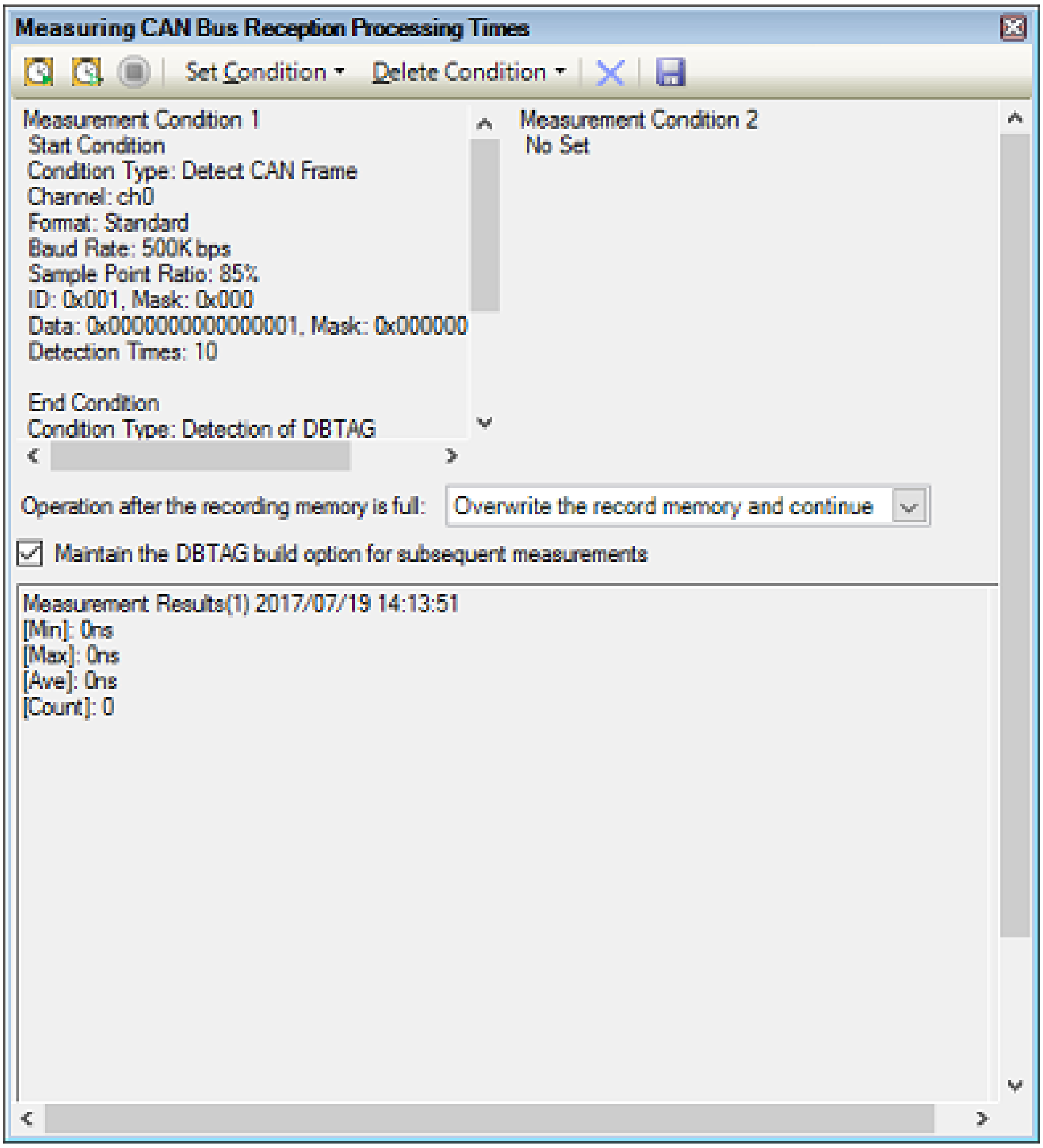
You can use this facility to reduce the number of steps required to measure the time from detection of a CAN frame on the bus to execution of the corresponding code in a program in the development of systems that use CAN communications.

[How to use]
This facility utilizes the following extended functions of the E2 emulator: CAN bus monitoring and time measurement. Accordingly, to use this facility, be sure to select [Use the power supplied from the target] for the [Interface for supplying the power] property under the [E2 Expansion Interface] category [E2] on the [Connect Settings] tab in the Property panel before connecting the debug tool.

Remark For details on the given extended functions of the E2 emulator, refer to the Application Note on the CAN Communication Time Measurement Solution (E2 Emulator, CS+).

(1) Open the Measuring CAN Bus Reception Processing Times panel
Open the Solution List panel and click on the [Measuring CAN Bus Reception Processing Times] button for measuring CAN bus reception processing times. The Measuring CAN Bus Reception Processing Times panel [E2] will open.

Figure 2.111 Measuring CAN Bus Reception Processing Times Panel

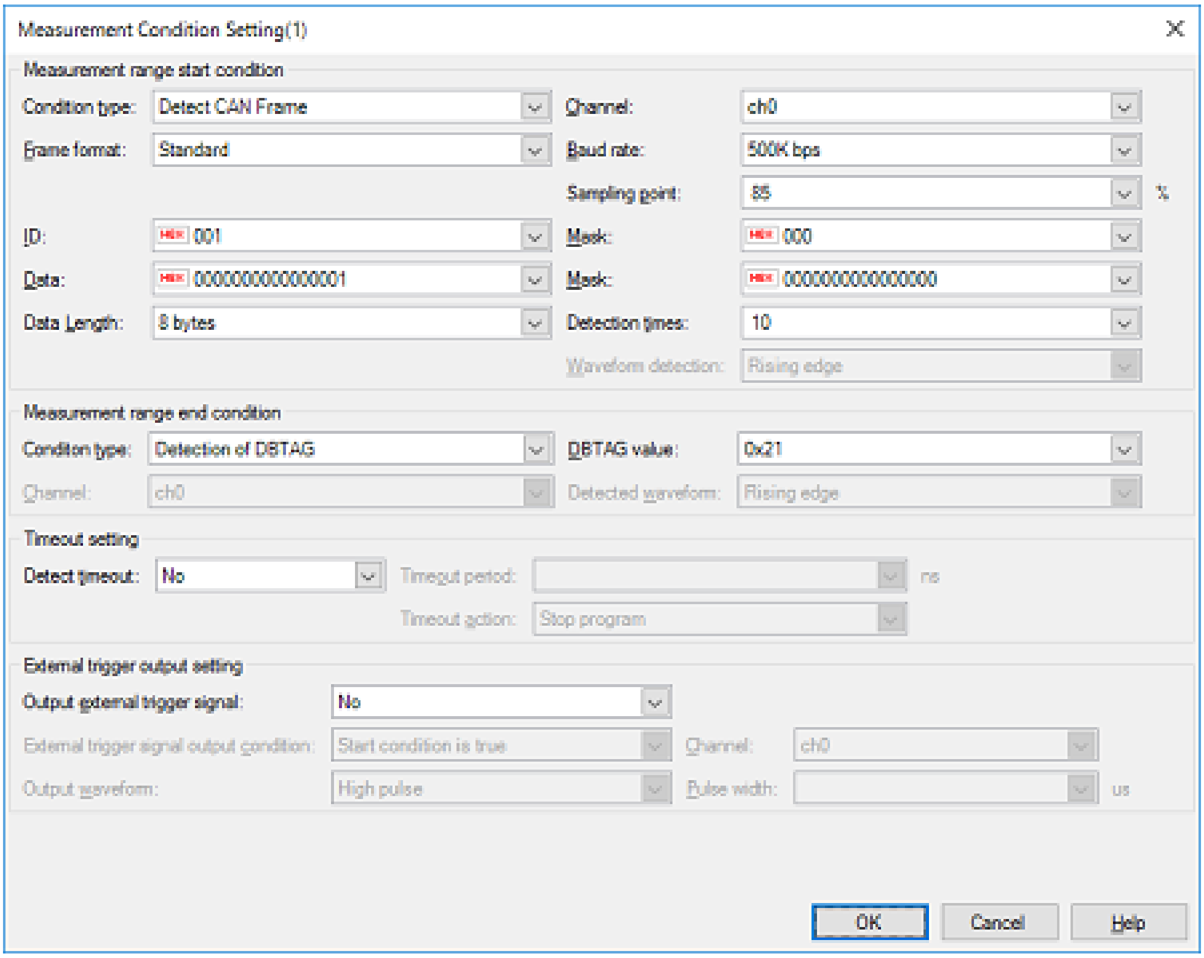


Remark The [Measuring CAN Bus Reception Processing Times] button is not displayed in the Solution List panel for devices that do not support the measuring CAN bus reception processing times.
Even if a device supports the measurement of current consumption, the [MMeasuring CAN Bus Reception Processing Times] button will be disabled when no debug tool is connected or an emulator other than the E2 emulator.

(2) Set the measurement conditions

You can select up to two conditions (conditions 1 and 2) since two timer channels are available for use in measurement.
To set the conditions, use the Measurement Condition Setting dialog box [E2]. Open this by selecting [Set Condition] -> [Set Condition 1...] or [Set Condition 2...] from the toolbar of the Measuring CAN Bus Reception Processing Times panel [E2].

Figure 2.112 Measurement Condition Setting Dialog Box



(a) Set a measurement-range start condition
The [Measurement range start condition] area allows you to select a condition that defines the beginning of the range over which measurement will proceed. You can measure the time from the beginning to the end of the range, respectively defined by the start and end conditions.
Select "Detect CAN Frame" or "Detect External Trigger Input Signal" for [Condition type]. Other items for which you need to make settings depend on the selected condition type.

<1> When "Detect CAN Frame" is selected
The beginning of the measurement range is a point where a specific CAN frame is detected by CAN bus monitoring, which is an extended function of the E2 emulator. You need to enter the following information regarding the CAN frame to be detected.

- Channel
  Select the channel for use in the detection of CAN frames by CAN bus monitoring through the E2 expansion interface.
- Frame format
  Select "Standard" or "Extended" as the format of CAN frames to be detected.
- Baud rate
  Select one of the following values as the bit rate for use in the detection of CAN communications.
  1M bps, 500K bps, 250K bps, 125K bps
- Sampling point
  Specify the relative position as a percentage (1 to 100%) within one bit period for the sampling of each bit of data in the CAN frame to be detected.

- ID and the ID mask value
  Specify ID in CAN frames to be detected and the mask value to be used as hexadecimal values.
  When 0 is selected for a mask bit of the mask value, the bit is treated as being masked.
- Data and the data mask value
  Specify data in CAN frames to be detected and the mask value to be used as hexadecimal values.
  When 0 is selected for a mask bit of the mask value, the bit is treated as being masked.
- Data Length
  Select the data size in bytes for CAN frames to be detected as a value from 0 to 8.
- Detection times
  Time measurement starts when CAN frames have been detected the number of times specified in this field.

<2> When "Detect External Trigger Input Signal" is selected
The beginning of the measurement range is a point where the input of an external trigger signal through the E2 expansion interface is detected. You need to enter the following information regarding the external trigger input signal to be detected.

- Channel
  Select the channel for use in detecting the input of external trigger signals through the E2 expansion interface.
- Waveform detection
  Select the type of waveform to be detected as the external trigger input from among "Rising edge", "Falling edge", and "Both edges", which are in the drop-down list.

(b) Set a measurement-range end condition
The [Measurement range end condition] area allows you to select conditions that define the end of the range over which measurement will proceed. You can measure the time from the beginning to the end of the range, respectively defined by the start and end conditions.
Select "Detection of DBTAG" or "Detection of external input trigger signal" for [Condition type]. Other items for which you need to make settings depend on the selected condition type.

<1> When "Detection of DBTAG" is selected
The end of the measurement range is a point where the execution of a dbtag instruction (part of the instruction set of RH850 MCUs) is detected. You need to select the value of DBTAG to be detected from among the following values.
0x21, 0x29, 0x31, 0x39, 0x41, 0x49, 0x51, 0x59, 0x61, 0x69

Go to "(3) Insert dbtag instructions at desired positions", insert dbtag instructions at desired positions, to select the value of DBTAG to be detected.

<2> When "Detection of external input trigger signal" is selected
The end of the measurement range is a point where the input of an external trigger signal through the E2 expansion interface is detected. You need to enter the following information regarding the external trigger input signal to be detected.

- Channel
  Select the channel for use in detecting the input of external trigger signals through the E2 expansion interface.
- Detected waveform
  Select the type of waveform to be detected as the external trigger input from among "Rising edge", "Falling edge", and "Both edges", which are in the drop-down list.

(c) Make timeout settings
The [Timeout setting] area allows you to make timeout settings. Use [Detect timeout] to select whether to enable timeout detection. With timeout detection enabled, CS+ will detect timeout in the form of the time entered in [Timeout period] having elapsed after time measurement was started but before the measurement-end conditions have been satisfied. Select one of the following actions in response to timeout detection.

- Detection only
  CS+ only detects the timeout and will not take any other action.
  Detection of timeout is only used as a condition for external trigger output in "(d) Make settings for the output of an external trigger signal".
- Stop internal tracing
  Tracing within the MCU stops.

- Stop program
  Execution of the program stops.

Caution "Stop internal tracing" is not selectable as the action on timeout detection when "Detection of DBTAG" is selected as the type of measurement-end condition.

(d) Make settings for the output of an external trigger signal
The [External trigger output setting] area allows you to make settings for the output of an external trigger signal.
Use [Output external trigger signal] to select whether to enable the output of an external trigger signal. With the output of an external trigger signal enabled, CS+ will output an external trigger (high-level pulse) signal when the required condition is satisfied.
Enter the following information.

<1> External trigger signal output condition
Select the condition for the output of an external trigger signal from among the following.

- Start condition is true
- End condition is true
- Timeout condition is true

Caution "Start condition is true" is not selectable when "Detect External Trigger Input Signal" is selected as the type of measurement-start condition.
"End condition is true" is also not selectable when "Detect External Trigger Input Signal" is selected as the type of measurement-end condition.
"Timeout condition is true" is not selectable when timeout detection is disabled.

<2> Channel
Select the channel for output of the external trigger through the E2 expansion interface.

<3> Pulse width
Specify the width of the high-level pulse to be output as the external trigger.

(3) Insert dbtag instructions at desired positions
Insert dbtag instructions at positions where you wish to detect their execution.
You need to specify a DBTAG value for each dbtag instruction. Select one of the following ten values.
0x21, 0x29, 0x31, 0x39, 0x41, 0x49, 0x51, 0x59, 0x61, 0x69

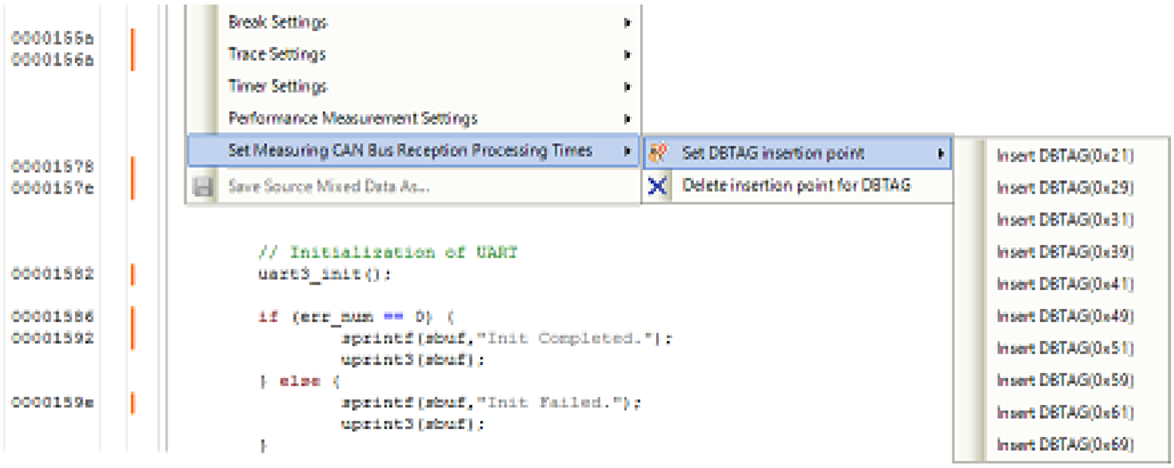
Remark See "RH850G3M/G3MH/G3K/G3KH User's Manual: Debug Instructions" for details on dbtag instructions.

Prior to measurement, you can insert dbtag instructions at desired positions and proceed with automatic rebuilding and downloading when V1.06.00 or a later version of the CC-RH compiler is selected for the active project.

Caution When you insert dbtag instructions prior to measurement and the [CPU Reset after download] property is set to [Yes] in the [Download] category on the [Download File Settings] tab in the Property panel, note that the CPU will be reset by rebuilding and downloading.

You can use the context menu in the Editor panel to set DBTAG insertion points at desired positions in the source code registered in the active project.

Figure 2.113 Context Menu in the Editor Panel



(4) Start measurement
Start measuring CAN bus reception processing times by using the facilities of the E2 emulator.
Click on or on the toolbar in the Measuring CAN Bus Reception Processing Times panel [E2].

Figure 2.114 Toolbar in the Measuring CAN Bus Reception Processing Times Panel

Caution CAN bus reception processing times are not measured by clicking on in the Main window.

(a) Clicking on
The project is rebuilt and downloaded before measurement starts. However, if the compiler used in the active project is a version of CC-RH earlier than V1.06.00, rebuilding and downloading do not proceed.
When [Maintain the DBTAG build option for subsequent measurements] in the Measuring CAN Bus Reception Processing Times panel [E2] is selected, the options which are specified for rebuilding and downloading before measurement are reflected in the [Parameters of software trace (DBTAG) for measuring CAN bus reception processing time] 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 CAN bus reception processing times without resets due to rebuilding and downloading.
Note that this button is not displayed if the compiler used in the active project is a version of CC-RH earlier than V1.06.00.

(5) 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 timeout, or by clicking on .

(6) View and save the results of measurement
Measurement ends at the same time as execution of the program is stopped.

(a) View the results of measurement
At the same time as measurement ends, the minimum time, maximum time, average time, and number of rounds of measurement are displayed in the measurement results display area of the Measuring CAN Bus Reception Processing Times panel [E2].
Check that the values shown are as intended.

(b) Save the results of measurement
Click on on the toolbar in the Measuring CAN Bus Reception Processing Times panel [E2] to save the latest results of measurement in a file.
Details of the CAN frames and dbtag instructions detected during measurement are also recorded with timestamps in the same file as the values displayed in the measurement results display area.

A. WINDOW REFERENCE

Appendix A provides detailed explanations of windows/panels/dialog boxes used for debugging with CS+.

A.1 Description

Windows/panels/dialog boxes for debugging are listed below.

Table A.1 Window/Panel/Dialog Box List

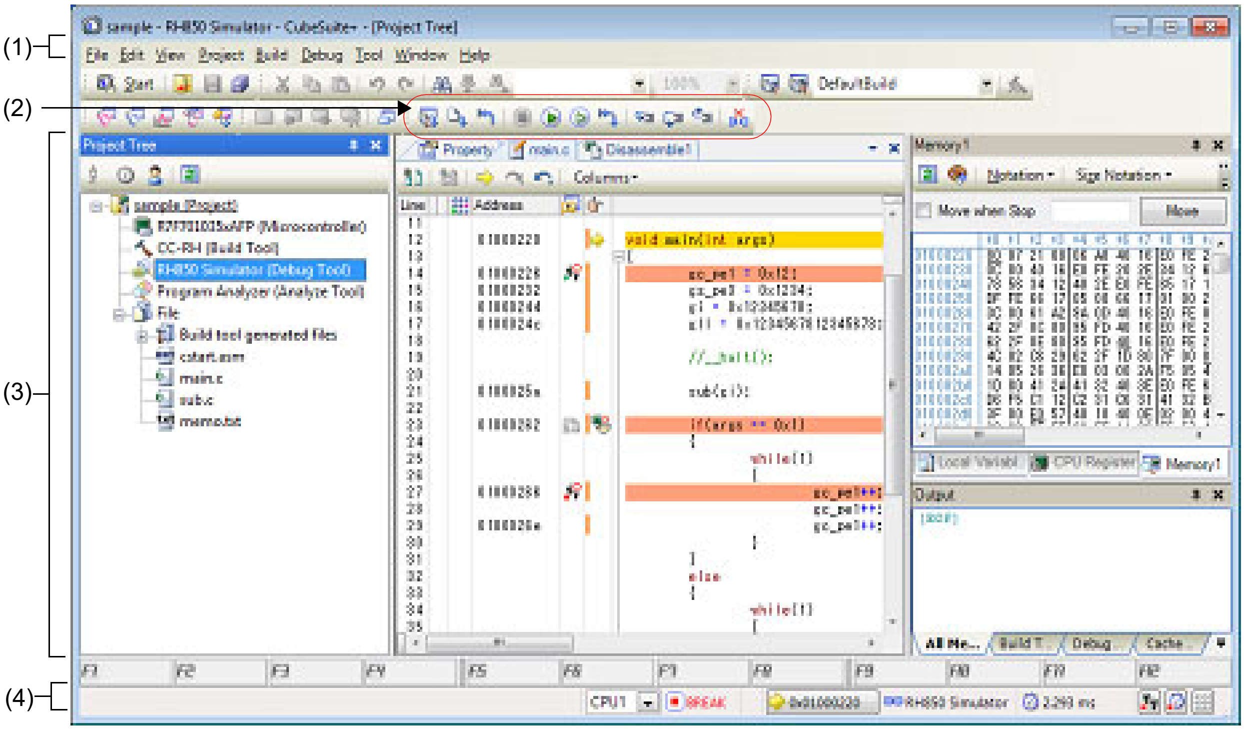
Window/Panel/Dialog Box Name	Description
Main window	Controls the program execution. Various windows, panels and dialog boxes can be opened from this window.
Debug Manager panel	Selects a core (PE) to be debugged and displays the core status.
Project Tree panel	Selects the debug tool to use.
Property panel	Displays detailed information on the debug tool currently selected in the Project Tree panel, and enables the settings of the tool to be changed.
Memory panel	Displays and modifies memory values.
Disassemble panel	Displays the results of memory value disassemble and is used to execute line assemble and instruction level debug.
CPU Register panel	Displays the contents of CPU registers, and modifies register values.
IOR panel	Displays and modifies I/O register values.
Local Variables panel	Displays and modifies local variables.
Watch panel	Displays and modifies registered watch-expression values.
Call Stack panel	Displays call stack information on function calls.
Trace panel	Displays trace data acquired from the debug tool.
Events panel	Displays detailed information on set events, switches the events between enabled and disabled, or deletes them.
Output panel	Displays messages output from the build tool/debug tool/plug-ins, or the results of batch searches carried out using the Find and Replace dialog box.
Select I/O Modules dialog box	Sets the I/O modules which details are displayed.
Select Debug target dialog box	Sets the debug target which details are displayed.
Select Contexts on Debug target dialog box	Sets the contexts on the debug target which details are displayed.
Select SPID filter dialog box	Sets the SPID filter which details are displayed.
Memory Mapping dialog box	Displays the memory mapping.
Download Files dialog box	Selects files to be downloaded and sets the download conditions.
Flash Options Setting dialog box	Configures options for the flash memory.
Action Events dialog box	Sets action events.
Column Number Settings dialog box	Specifies the number of view columns of memory values on the Memory panel.
Address Offset Settings dialog box	Specifies an offset value for the address display on the Memory panel.
Memory Initialize dialog box	Initializes memory.

Window/Panel/Dialog Box Name	Description
Memory Search dialog box	Searches memory.
Print Address Range Settings dialog box	Sets the address range to print the contents of the Disassemble panel.
Trace Search dialog box	Searches trace data.
Detail dialog box (for execution events)	Displays and modifies the detailed information on an execution-related event.
Detail dialog box (for access events)	Displays and modifies the detailed information on an access-related event.
Detailed Settings of Point Trace dialog box	Displays and modifies the detailed information on the point trace event.
Detailed Settings of Timer Measurement dialog box [Full-spec emulator][E1][E20]	Displays and modifies the detailed information on the timer event.
Detailed Settings of Performance Measurement dialog box [Full-spec emulator][E1][E20]	Displays and modifies the detailed information on the Performance Measurement event.
Scroll Range Settings dialog box	Sets the scroll range for the Memory panel/Disassemble panel.
Go to the Location dialog box	Moves the caret to the specified position.
Data Save dialog box	Saves the settings and other data displayed in the respective windows/panels/dialog boxes or saves upload data.
Specified Section dialog box	Specifies the range for skipping step execution.
Functions and Variables Access Table panel	Displays the functions that access variables in the form of an orthogonal table.
Exclusive Control Checking Tool dialog box	Checks whether exclusive control is correctly performed for a certain variable and makes the settings required for checking.
Pseudo-Error Debugging panel [Full-spec emulator][E1][E20]	This panel is central to the functionality of the solution for pseudo-error debugging.
Select Pseudo-Error dialog box [Full-spec emulator][E1][E20]	Selects ECM pseudo-errors displayed in the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].
Breakpoint Setting dialog box [Full-spec emulator][E1][E20]	Sets the breakpoints that are to be registered to the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].
Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20]	This panel is central to the functionality of the solution for debugging of CAN bus reception.
RS-CAN Module Setting dialog box [Full-spec emulator][E1][E20]	Makes settings related to the entire RS-CAN module.
Receive Channel Setting dialog box [Full-spec emulator][E1][E20]	Makes settings related to the receive channel.
Transmit Channel Setting dialog box [Full-spec emulator][E1][E20]	Makes settings related to the transmit channel.
Transmit Frame Setting dialog box [Full-spec emulator][E1][E20]	Makes settings related to the transmit frame.
Measuring CAN Bus Reception Processing Times panel [E2]	This panel is central to the functionality of the solution for measurement of CAN bus reception processing times.
Measurement Condition Setting dialog box [E2]	This dialog box is used to set conditions for measurement in the Measuring CAN Bus Reception Processing Times panel [E2].

Main window

This window is automatically opened when CS+ is started up.
In this window, you can control the program execution and open panels for the debugging process.

Figure A.1 Main Window



This section describes the following.

- [How to open]
- [Description of each area]

[How to open]

- From the Windows [Start] menu, select [All Programs] >> [Renesas Electronics CS+] >> [CS+ for CC].

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) Menubar
Menu items related to the debugging are described below.

Remark The items that can be selected in each menu can be customized using the User Setting dialog box.

(a) [View]
The [View] menu provides the following items and functions (default).

Debug Manager	Opens the Debug Manager panel This item is disabled when the selected microcontroller version does not support multi-core or when disconnected from the debug tool.

Watch	The following cascade menus are displayed to open the Watch panel. These items are disabled when 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.
Call Stack	Opens the Call Stack panel.
Memory	The following cascade menus are displayed to open the Memory panel. These items are disabled when 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. This item is disabled when disconnected from the debug tool.
CPU Register	Opens the CPU Register panel. This item is disabled when disconnected from the debug tool.
Trace	Opens the Trace panel. This item is disabled when disconnected from the debug tool.
Disassemble	The following cascade menus are displayed to open the Disassemble panel. These items are disabled when 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. This item is disabled when disconnected from the debug tool.
Show Current PC Location	Displays the current PC position in the Editor panel. This item is disabled when disconnected from the debug tool.
Back to Last Cursor Position	Goes back to the position before jumping (see "2.7.2.4 Move to the symbol defined location") to the defined location. This item is disabled when disconnected from the debug tool.
Forward to Next Cursor Position	Forwards to the position before operating [Back to Last Cursor Position].
Tag Jump	Jumps to the corresponding line/column in the corresponding file if the information of a file name/line number/column number exists in the line at the caret position on the Editor panel/Output panel.

(b) [Debug]

The [Debug] menu provides the following items and functions (default).

Debug Solutions	A cascade menu is displayed to open a window of solutions related to debugging functions.

Download	Downloads the specified file(s) into 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 build 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+.
Upload...	Opens the Data Save dialog box to save the memory contents. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool.
Disconnect from Debug Tool	Disconnects from the currently connected debug tool. This item is disabled during program execution/build (not including rapid build) execution or when 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.
RH850 IE850A	Uses IE850A as the debug tool.
RH850 Full-spec emulator	Uses Full-spec emulator as the debug tool.
RH850 E2	Uses E2 as the debug tool.
RH850 E1(LPD)	Uses E1 in LPD communication mode as the debug tool.
RH850 E20(LPD)	Uses E20 in LPD communication mode as the debug tool.
RH850 Simulator	Uses Simulator as the debug tool.
Stop	Forcibly stops the program currently being executed. This item is disabled when the program is already halted or disconnected from the debug tool.
Go	Executes the program from the current PC position. Execution of the program will be stopped when the condition of a set break event is met. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool.

Ignore break and go	Executes the program from the current PC position. Execution of the program continues, ignoring set break events and action events. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool.
Step In	Executes the program step by step[Note] from the current PC position (Step in execution). However, in the case of a function call, the program is stopped at the beginning of the function having been called. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool.
Step Over	Executes the program step by step[Note] from the current PC position (Step over execution). In the case of a function call by the jarl instruction, all the source lines/instructions in the function are treated as one step and executed until the position where execution returns from the function (step-by-step execution will continue until the same nest is formed as when the jarl instruction has been executed). In the case of an instruction other than jarl, operation is the same as when [Step In] is selected. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool.
Return Out	Executes the program until execution returns from the current function (or returns to the calling function)[Note](Return out execution). This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool.
CPU Reset	Resets the CPU (does not execute a program) This item is disabled during build (not including rapid build) execution or when disconnected from the debug tool.
Restart	Resets the CPU and then executes the program from the reset address. This item is disabled during build (not including rapid build) 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. For details on the features and usage, see “Saving and Restoring the States of Debug Tools”.
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. For details on the features and usage, see “Saving and Restoring the States of Debug Tools”.
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 carried out either in units of source lines or in units of instructions.
For details, see "2.10.3 Execute programs in steps".

(2) Debug toolbar
The debug toolbar includes the buttons that control the execution of programs.
The debug toolbar provides the following buttons and functions (default).

Remark 1. The buttons on the toolbar can be customized using the User Setting dialog box. Furthermore, a new toolbar can be created using the same dialog box.

Remark 2. A Group of toolbar displayed can be selected with the context menu that is displayed by right-clicking on the toolbar.

	Executes the build of a project and downloads the file into 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. When the build has failed, download will not be executed. The function of this item is the same as that of [Build & Download] in the [Debug] menu.
	Downloads the specified file(s) into 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. The function of this item is the same as that of [Download] in the [Debug] menu.
	Resets the CPU (does not execute a program) This item is disabled during build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [CPU Reset] in the [Debug] menu.
	Forcibly stops the program currently being executed. This item is disabled when the program is already halted or disconnected from the debug tool. The function of this item is the same as that of [Stop] in the [Debug] menu.
	Executes the program from the current PC position. Execution of the program will be stopped when the condition of a set break event is met. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Go] in the [Debug] menu.
	Executes the program from the current PC position. Execution of the program continues, ignoring set break events and action events. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Ignore break and go] in the [Debug] menu.
	Resets the CPU and then executes the program from the reset address. This item is disabled during build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Restart] in the [Debug] menu.
	Executes the program step by step[Note] from the current PC position (Step in execution). However, in the case of a function call, the program is stopped at the beginning of the function having been called. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Step In] in the [Debug] menu.
	Executes the program step by step[Note] from the current PC position (Step over execution). In the case of a function call by the jarl instruction, all the source lines/instructions in the function are treated as one step and executed until the position where execution returns from the function (step-by-step execution will continue until the same nest is formed as when the jarl instruction has been executed). In the case of an instruction other than jarl, operation is the same as when the button is clicked. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Step Over] in the [Debug] menu.
	Executes the program until execution returns from the current function (or returns to the calling function)[Note] (Return out execution). This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Return Out] in the [Debug] menu.

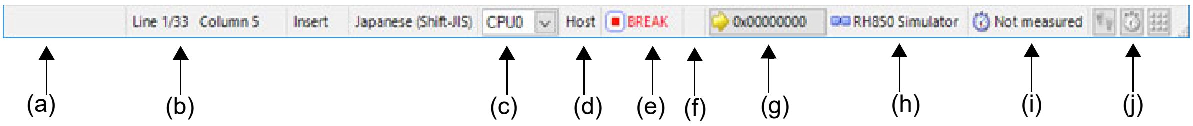
	Disconnects from the currently connected debug tool. This item is disabled during program execution/build (not including rapid build) execution or when disconnected from the debug tool. The function of this item is the same as that of [Disconnect from Debug Tool] in the [Debug] menu.

Note Step execution can be carried out either in units of source lines or in units of instructions.
For details, see "2.10.3 Execute programs in steps".

(3) Panel display area
This area displays the various panels.
For details on the display content, see the sections describing the individual panels.

(4) Statusbar
Statusbar displays the following items of information.

Figure A.2 Statusbar



(a) Status message
This area displays the following messages and other information.

- A brief explanation of the selected menu item
- A message reporting that an invalid value has been input in the panel/dialog
- A message reporting that the specified character string has not been found as a result of a search using the Find and Replace dialog box
- A statement of the cause of the break when a break has occurred (see "2.11 Stop Programs (Break)")

(b) Focus panel status information
This area displays status information on the panel currently having the focus.
Note that nothing is displayed here for a panel that has no status information.

(c) Selection of debug target core
This area is used to select a core (PE) to be debugged (see "2.9 Select a Core (PE)").
Note that nothing is displayed here when the selected microcontroller version does not support multi-core or when disconnected from the debug tool.

Caution When the drop-down list used for switching between cores on the status bar is being displayed while the size of this window is maximized, part of the list is hidden behind the task bar and thus cannot be selected.
Set the task bar to "Hide automatically" or set the location of the task bar as [Right], [Left], or [Upper].

(d) Current context
This area displays the current context of the core (PE) being debugged.
However, the state is not displayed if the microcontroller does not have information on the context or the debug tool is not connected.

Current context	Item Displayed
Undefined since the program is being executed	Run
Host mode	Host
Guest mode (GPID*x*: *x* is any number from 0 to 7.)	Guest(GPID*x*)

(e) Running state
This area displays the state of the program with the following icons and character strings.
Note that nothing is displayed here when the debug tool is not connected.

State of Program	Displayed Content
Under execution	RUN
Now halted	BREAK
Step execution in progress	STEP

(f) CPU status
This area displays the current CPU status of the debug tool. When there is the possibility that the CPU is in two or more statuses, the corresponding display contents are displayed separated by "&".
Note that nothing is displayed here when the debug tool is not connected.

Debug Tool	Displayed Content	CPU Status
Full-spec emulator E1/E20	Halt	In HALT mode
	Stop	In STOP mode
	Reset	In reset state
	Pow Off	Power not supplied to the target
	Initial Stop	In initial stop state
	Deep Stop	In Deep Stop mode
	Cyclic Run	In Cyclic Run mode
	Cyclic Stop	In Cyclic Stop mode
	Cyclic Disable	This is the state of the core other than the main core when the main core is in the Cyclic Run or Cyclic Stop mode.
	Standby	A clock signal is not being supplied to the GTM.
	Disable	The selected multi-channel sequencer (MCS) is not activated.
Simulator	Halt	In HALT mode
	Stop	In STOP mode
	Reset	In reset state

(g) Current PC position
This area displays the current PC position with a hexadecimal value. When this area is clicked, the caret moves to the current PC position on the Editor panel.
In addition, when the mouse pointer is placed over this area, a pop-up window appears to display the following information: "Current PC: 0x *current PC value* (*source name#line count*[Note])".
Note that nothing is displayed here when the debug tool is not connected.

Note "*symbol name+offset value*" is displayed when acquisition of information is impossible.

Remark "Running" is displayed in this area during execution of a program.

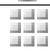
(h) Connection state
This area displays the current state of connection with the debug tool using the following icons and character strings.

Connection State	Displayed Content
Connected	*Debug tool name*
Disconnected	DISCONNECTED

(i) Run-Break Timer measurement result
This area displays the result of measurement by the Run-Break Timer event (the unit of value used differs depending on the measurement amount). See "2.15.1 Measure execution time until stop of the execution".
Note that nothing is displayed here when the debug tool is not connected.

Condition	Displayed Content
Un-measuring	Not measured
Under measurement	Measuring
When a timer measurement overflow has occurred	OVERFLOW

(j) Debug tool state
This area displays the current state of debug tool's functions using the following icons and character strings. Note that nothing is displayed here when the debug tool is not connected.

Function	Use		Not Use
	Being Executed	Stopped	
Trace			
Timer			
Coverage			

Remark **[Simulator]**
When the program is halted, clicking the appropriate icon enables the state to be switched between "Use" and "Not use". The result of switching will be reflected in the setting of the [Use trace function]/[Use timer function]/[Use coverage function] property in the [Trace]/[Timer]/[Coverage] category on the [Debug Tool Settings] tab of the Property panel.

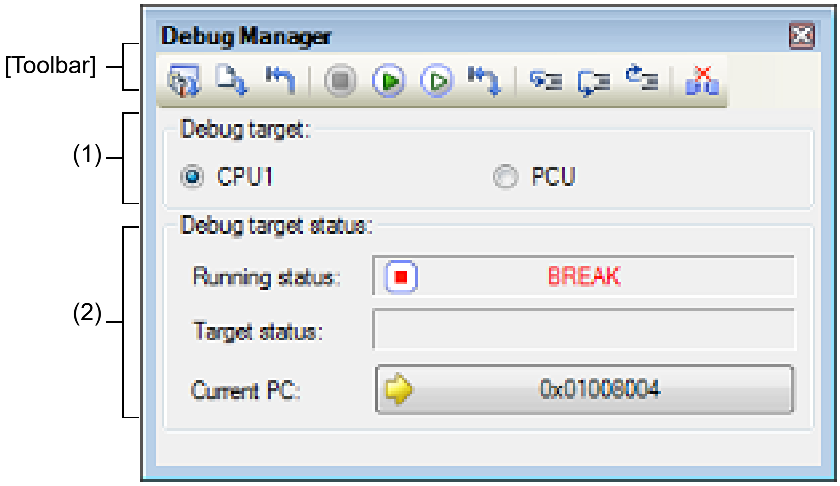
Debug Manager panel

When the selected microcontroller is the multi-core product, this panel is used to select a core (PE: Processer Element) to be debugged and display the core status (see "2.9 Select a Core (PE)").

This panel appears only when connected to the debug tool.

Caution This panel cannot be opened when the selected microcontroller is the single-core product.

Figure A.3 Debug Manager Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]

[How to open]

- From the [View] menu, select [Debug Manager].

[Description of each area]

(1) [Selects debug target core] area
Select a core (PE) to be debugged with a option button.

Remark 1. [Simulator]
This area becomes invalid during execution of a program.

Remark 2. You can also select a core to be debugged on the statusbar in the Main window.

(2) [Debug target core status] area
This area displays the status of the core currently being selected.

Remark You can also confirm the information displayed in this area on the statusbar in the Main window.

(a) [Running status]
Displays the current state of the program with the following icons and character strings.

State of Program	Displayed Content
Running	RUN
Stopped	BREAK
In step execution	STEP

(b) [Core status]
Displays the current core statuses of the debug tool. When there is the possibility that the core is in two or more statuses, the corresponding display contents are displayed separated by "&".

Debug Tool	Displayed Content	Core Status
Full-spec emulator E1/E20	Halt	In HALT mode
	Stop	In STOP mode
	Reset	In reset state
	Pow Off	Power not supplied to the target
	Initial Stop	In initial stop state
	Deep Stop	In Deep Stop mode
	Cyclic Run	In Cyclic Run mode
	Cyclic Stop	In Cyclic Stop mode
	Cyclic Disable	This is the state of the core other than the main core when the main core is in the Cyclic Run or Cyclic Stop mode.
	Standby	A clock signal is not being supplied to the GTM.
	Disable	The selected multi-channel sequencer (MCS) is not activated.
Simulator	Halt	In HALT mode
	Stop	In STOP mode
	Reset	In reset state

(c) [Current PC]
Displays the current PC position with a hexadecimal value. When this button is clicked, the caret moves to the current PC position on the Editor panel.

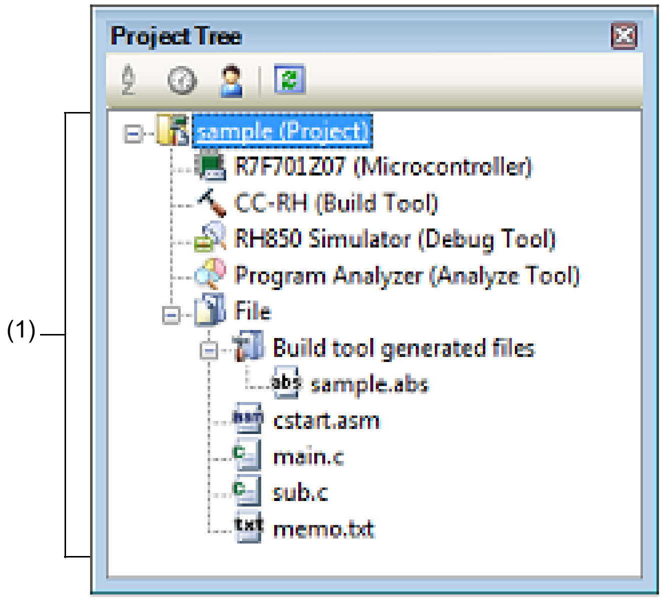
[Toolbar]

The function of this toolbar is the same as that of the Debug toolbar on the Main window. For details on the function of each button, see "(2) Debug toolbar".

Project Tree panel

This panel is used to display the project components (Microcontroller, Build Tool, Debug Tool, etc.) in a tree structure. On this panel, you can select or change the debug tool to use.

Figure A.4 Project Tree Panel



The following items are explained here.

- [How to open]
- [Description of each area]
- [Context menu]

[How to open]

- From the [View] menu, select [Project Tree].

[Description of each area]

(1) Project tree area
Project components are displayed in tree view with the following given node.

Node	Description
Microcontroller type Debug tool name (Debug tool)	- *Microcontroller type*: The selected microcontroller type is displayed. - *Debug tool name*: The debug tool (IE850A, Full-spec emulator, E2, E1(LPD), E20(LPD), or Simulator) currently being used in the project is displayed[Note]. Simulator is selected when a new project is created.

Note The selectable debug tools differ depending on the microcontroller selected in the project.

Select the debug tool node to configure with the Property panel. If the Property panel is not being opened, double-click the node to open the corresponding Property panel.

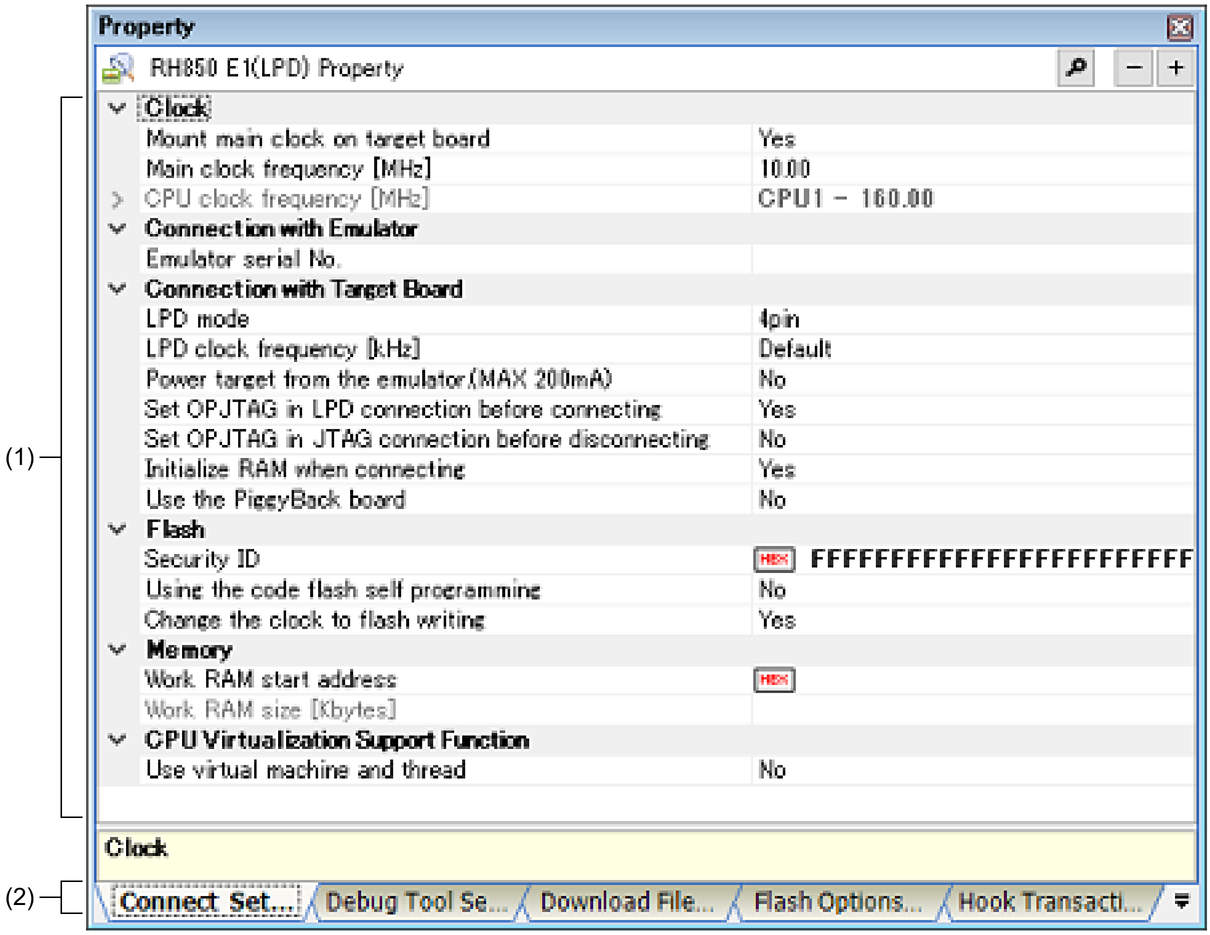
[Context menu]

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.
RH850 IE850A	Uses IE850A as the debug tool.
RH850 Full-spec emulator	Uses Full-spec emulator as the debug tool.
RH850 E2	Uses E2 as the debug tool.
RH850 E1(LPD)	Uses E1 in LPD communication mode as the debug tool.
RH850 E20(LPD)	Uses E20 in LPD communication mode as the debug tool.
RH850 Simulator	Uses Simulator as the debug tool.
Property	Displays the selected category node's property in the Property panel.

Property panel

This panel is used to display and set the debug tool operation environment that is selected in the Project Tree panel.

Figure A.5 Property Panel (When E1 Is Selected)



This section describes the following.

- [How to open]
- [Description of each area]
- [[Edit] menu (Property panel-dedicated items)]
- [Context menu]

[How to open]

- On the Project Tree panel, select the [*Microcontroller type Debug tool name* (Debug Tool)] node to use, and then select [Property] from the [View] menu or the context menu.
- On the Project Tree panel, double-click the [*Microcontroller type Debug tool name* (Debug Tool)] node to use.

Remark If this panel has been opened, the detailed information on the debug tool is displayed by selecting the [*Microcontroller type Debug tool name* (Debug Tool)] node on the Project Tree panel.

[Description of each area]

(1) Detailed information display/change area
In this area, the detailed information on the debug tool that is selected in Project Tree panel is displayed by category in the list. Also, you can directly change its settings.

The ⊟ mark is indicates all the items in the category are expanded. The ⊞ mark indicates all the items are collapsed. You can expand/collapse the items by clicking these marks or double-clicking the category name.
Note that only the hexadecimal number is allowed in the text box if the HEX mark is displayed in the property configuration area.
For details on the information/how to setup in the category and property items contained in it, see the section explaining the corresponding tab.

(2) Tab selection area
Categories for the display of the detailed information are changed when each tab is selected.
In this panel, following tabs are contained (see the section explaining each tab for details on the display/setting on the tab).

- [Connect Settings] tab
- [Debug Tool Settings] tab
- [Download File Settings] tab
- [Flash Options Settings] tab
- [Hook Transaction Settings] tab

[[Edit] menu (Property panel-dedicated items)]

Undo	Undoes the latest property value editing being done.
Cut	Deletes the selected character string(s) and copies them to the clipboard while editing the property value.
Copy	Copies the contents of the selected range to the clipboard as character string(s).
Paste	Pastes the contents of the clipboard to the property value while editing the property value.
Delete	Deletes the selected character string(s) while editing the property value.
Select All	Selects all the character strings in the selected property while editing the property value.
Find...	Opens the Find and Replace dialog box with selecting [Quick Find] tab.

[Context menu]

[While not editing the property value]

Reset to Default	Restores the selected setting of the property item to default value.
Reset All to Default	Restores all the selected settings of the property items on the tab to default value.

[While editing the property value]

Undo	Undoes the latest property value editing being done.
Cut	Deletes the selected character string(s) and copies them to the clipboard while editing the property value.
Copy	Copies the contents of the selected range to the clipboard as character string(s).
Paste	Pastes the contents of the clipboard to the property value while editing the property value.
Delete	Deletes the selected character string(s) while editing the property value.
Select All	Selects all the character strings in the selected property while editing the property value.

[Connect Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.

(1) [Clock]
(2) [Connection with Emulator] [E1] [E20]
(3) [Connection with Target Board] [Full-spec emulator][E1][E20]
(4) [Multi-core] [E1] [E20]
(5) [External Trace] [IE850A]
(6) [E2 Expansion Interface] [E2]
(7) [Flash] [Full-spec emulator][E1][E20]
(8) [Memory]
(9) [Hardware-assisted Virtualization]
(10) [Configuration] [Simulator]
(11) [CPU Virtualization Support Function]

[Description of each category]

(1) [Clock]
The detailed information on clocks is displayed and its configuration can be changed.

<table>
<tr><td rowspan="5">Mount main clock on target board</td><td colspan="3">Select whether to mount the main clock on the target board.
Select [No] when you use the on-chip oscillation circuit instead of the main clock circuit.
When [No] is selected, the following properties become fixed values.
[LPD mode]: [4pin]
[Set OPJTAG in LPD connection before connecting]: [No]</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Mounts the main clock.</td></tr>
<tr><td>No</td><td>Does not mount the main clock.</td></tr>
<tr><td rowspan="4">Main clock frequency [MHz]</td><td colspan="3">Specify the main clock frequency (before multiplication) in MHz unit.
Caution When the instruction simulator for RH850 is used, the CPU clock frequency will always be the same as the value of the main clock frequency set in this property.</td></tr>
<tr><td>Default</td><td colspan="2">[Full-spec emulator][E1][E20]
10.00
[Simulator]
[RH850G3M, RH850G3K, RH850G3MH, RH850G3KH] 320.00
[RH850G4MH, RH850G4KH] 400.00</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list or directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Either one of the following from the drop-down list
[Full-spec emulator][E1][E20]
10.00, 20.00 (unit: MHz)
[Simulator]
1.00, 2.00, 3.00, 3.57, 4.00, 4.19, 4.91, 5.00, 6.00, 7.20, 8.00, 8.38, 9.60, 10.00, 12.00, 16.00, 20.00, 25.00, 30.00, 32.00, 33.33, 34.00, 40.00, 48.00, 50.00, 64.00, 80.00, 160.00, 240.00, 320.00, 400.00 (unit: MHz)
Directly enter the numbers ranged below
0.001 to 999.999 (unit: MHz)</td></tr>
<tr><td>CPU clock frequency[MHz]
[Full-spec emulator]
[E1][E20]</td><td colspan="3">Specify the CPU clock frequency for each core.
The CPU clock frequency for each core can be specified with subproperties of this property. The CPU clock frequency is used to convert the time stamp information for a trace to an actual time.
The number of subproperties displayed differs with the selected microcontroller.</td></tr>
</table>

Core name (Subproperty) [Full-spec emulator] [E1][E20]	Displays the name of the core incorporated in the selected microcontroller.	
	Default	Depends on the selected microcontroller
	Modifying	Changes not allowed
CPU clock frequency (Subproperty) [Full-spec emulator] [E1][E20]	Specify the CPU clock frequency (after multiplication) of the *core name*.	
	Default	Depends on the selected microcontroller
	Modifying	Select from the drop-down list or directly enter from the keyboard.
	Available values	0.001 to 999.999 (unit: MHz)
Select Timer/Trace clock frequency [Simulator]	Displays the clock frequency for using timer/trace function.	
	Default	CPU clock frequency
	Modifying	Changes not allowed
Unit of Timer/Trace clock frequency [Simulator]	Displays the unit of the clock frequency for using timer/trace function.	
	Default	MHz
	Modifying	Changes not allowed
Timer/Trace clock frequency [Simulator]	Displays the value of the clock frequency for using timer/trace function. Note, however, that "---_---" is displayed while disconnected from the debug tool.	
	Default	320.00
	Modifying	Changes not allowed
Peripheral ultra high speed clock (CLK_UHSB) [MHz] [E2][IE850A]	Specify the frequency of the peripheral ultra-high-speed clock (CLK_UHSB) in MHz. The peripheral ultra-high-speed clock (CLK_UHSB) is used when information on the time stamps during tracing is to be converted to the actual times.	
	Default	160.00
	Modifying	Select from the drop-down list or directly enter from the keyboard.
	Available values	Either one of the following from the drop-down list 160.00, 80.00, 40.00 (unit: MHz) Directly enter the numbers ranged below 0.001 to 999.999 (unit: MHz)

(2) [Connection with Emulator] [E1] [E20]

Emulator serial No.	Select the serial No. of the emulator to be connected.[Note] The drop-down list is updated every time it is used.	
	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 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.

(3) [Connection with Target Board] [Full-spec emulator][E1][E20]
The detailed information on the connection to the target board is displayed and its configuration can be changed.

Caution Properties in this category cannot be changed when CS+ is connected to the debug tool.

Use emulation adapter [IE850A]	Select whether to use the emulation adapter.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Uses the emulation adapter.
		No	Does not use the emulation adapter.
Connecting with target board [Full-spec emulator]	Select whether the target board is connected to Full-spec emulator.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Target board is connected.
		No	Target board is not connected.
Communications method [IE850A][E2]	Select the method through which the emulator communicates with the CPU on the target system.		
	Default	LPD	
	Modifying	Select from the drop-down list. However, changeable only when disconnected from the debug tool	
	Available values	LPD, JTAG	
LPD mode [E1][E20]	Select LPD communication mode to be used.		
	Default	Depends on the selected microcontroller.	
	Modifying	Select from the drop-down list.	
	Available values	Depends on the selected microcontroller.	
Baud rate [kbps] [E1][E20]	Select the baud rate for LPD communication. This property appears only when the [LPD mode] property is set to [1pin].		
	Default	500	
	Modifying	Select from the drop-down list.	
	Available values	500, 1000, 2000 (unit: Kbps)	
LPD clock frequency [kHz] [E1][E20]	Specify the clock frequency for LPD communication. When [Default] is selected, the default value specific to the microcontroller is used in connection to the target board. This property appears only when the [Communications method] property is set [LPD] and [LPD mode] property is set to [4pin].		
	Default	Default	
	Modifying	Select from the drop-down list. However, changeable only when disconnected from the debug tool	
	Available values	[E1][E20] Default, 3000, 5500, 11000, 16500 (unit: kHz) [E2][IE850A] Default, 3000, 55500, 11000, 16500, 20000, 25000, 33000, 40000 (unit: kHz)	

<table>
<tr><td rowspan="4">JTAG clock fre-
quency [kHz]</td><td colspan="3">Specify the clock frequency for JTAG communication.
When [Default] is selected, the default value specific to the microcontroller is used in connection to the target board.
This property appears only when the [Communications method] property is set [JTAG].</td></tr>
<tr><td>Default</td><td colspan="2">Default</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.
However, changeable only when disconnected from the debug tool</td></tr>
<tr><td>Available values</td><td colspan="2">Default, 6250, 11000, 16500, 20000, 25000</td></tr>
<tr><td rowspan="5">Power target from the emulator (MAX 200mA)
[E1]</td><td colspan="3">Select whether to supply power to the target board from E1.
This item is not displayed when IE850A is selected.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Supplies power to the target board.</td></tr>
<tr><td>No</td><td>Does not supply power to the target board.</td></tr>
<tr><td rowspan="5">Interface for supply-
ing the power
[E2]</td><td colspan="3">Select the interface for supplying the power to the target board from the emulator.
This property appears only when the [Power target from the emulator (MAX 200mA)] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">USER I/F</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>USER I/F</td><td>Uses the user interface.</td></tr>
<tr><td>E2 expansion I/F</td><td>Uses the E2 expansion interface.</td></tr>
<tr><td rowspan="4">Supply voltage
[E1]</td><td colspan="3">Select the power voltage supplied to the target board.
This property appears only when the [Power target from the emulator (MAX 200mA)] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">3.3V</td></tr>
<tr><td>Modifying</td><td colspan="2">[E1]
Select from the drop-down list.
[E2]
Select from the drop-down list or directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">[E1]
3.3, 5.0
[E2]
Directly enter the numbers ranged below
1.8 to 5.0 (unit: V)</td></tr>
</table>

<table>
<tr><td rowspan="5">Set OPJTAG in LPD connection before connecting
[E1][E20]</td><td colspan="3">Select whether to start up the microcontroller in serial programming mode upon connection to the debug tool and change the option byte settings to select LPD connection.
This property appears when the selected microcontroller requires it.
This property is not displayed when the [Communications method] property has been displayed and [JTAG] was selected.
This property is also not displayed for devices that do not have internal ROM.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.
However, changeable only when disconnected from the debug tool</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Starts up the microcontroller in serial programming mode upon its connection to CS+. The debug tool then checks the OPJTAG byte and, if LPD is not selected, changes the setting to select LPD. After that, the microcontroller enters debugging mode (default).</td></tr>
<tr><td>No</td><td>Starts up the microcontroller in debugging mode upon its connection to CS+. The debug tool then checks OPJTAG and, if LPD is not selected, shows a message dialog box.</td></tr>
<tr><td rowspan="5">Set OPJTAG in JTAG connection before disconnecting
[E1][E20]</td><td colspan="3">Select whether to change the option byte settings to select JTAG connection before disconnection of the debug tool.
This property appears when the selected microcontroller requires it.
This property is not displayed when the [Communications method] property has been displayed and [JTAG] was selected.
This property is also not displayed for devices that do not have internal ROM.
When [No] (the default setting) is selected, the option byte settings are not changed before the debug tool is disconnected. In this case, LPD mode is applicable as the pin mode.

Remark On connection to E1, CS+ changes the option byte settings if LPD is not selected. For this reason, connecting and disconnecting E1 may change the value of the option bytes.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.
Note that changes can be made only when the [Set OPJTAG in LPD connection before connecting] property is set to [Yes].
However, changeable only when disconnected from the debug tool</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Changes the option byte settings to select JTAG connection before disconnection of the debug tool.</td></tr>
<tr><td>No</td><td>Does not change the option byte settings before disconnection of the debug tool.</td></tr>
</table>

<table>
<tr><td rowspan="5">Initialize RAM when connecting
[Full-spec emulator]
[E1][E20]</td><td colspan="3">Select whether to initialize the RAM when connecting to the debug tool.
Caution When [No] is selected, the following functions involving writing to the flash memory cannot be used.
- Downloading
- Writing from the Memory panel, Watch panel, Local Variables panel, or Disassemble panel
- Writing of the option bytes by using the Flash Options Setting dialog box
- Setting of software breakpoints
An ECC error may also occur during access to the RAM from the Memory panel, Watch panel, Local Variables panel, etc.
When debugging an ECC error that has occurred, do not open the Memory panel, Watch panel, Local Variables panel, etc. before the RAM has been initialized by the user program.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Initializes the RAM.</td></tr>
<tr><td>No</td><td>Does not initialize the RAM.</td></tr>
<tr><td rowspan="5">Enable security function when connecting
[Full-spec emulator]</td><td colspan="3">Select whether to enable the security function (ICU-S) when connecting to the debug tool.
Once the security function is enabled, it cannot be disabled after that.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Enable the security function (ICU-S) when connecting to the debug tool.</td></tr>
<tr><td>No</td><td>Does not enable the security function (ICU-S) when connecting to the debug tool.</td></tr>
<tr><td rowspan="5">Release the RESET before disconnecting from the target system
[IE850A][E2]</td><td colspan="3">Select whether to release the RESET before disconnecting from the target system.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Release the RESET before disconnecting from the target system.</td></tr>
<tr><td>No</td><td>Does not release the RESET before disconnecting from the target system.</td></tr>
<tr><td rowspan="5">Use the PiggyBack board
[E1][E20]</td><td colspan="3">Select whether to use the PiggyBack board.
Select [Yes] to use the PiggyBack board. The emulator may not started if a PiggyBack board is in use but [No] is selected.
This property only appears when the selected microcontroller requires it.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Uses the PiggyBack board.</td></tr>
<tr><td>No</td><td>Does not use the PiggyBack board.</td></tr>
</table>

Debug the initial stop state and the standby mode	Select whether debugging is to proceed in the initial stop state and in standby mode. For details on this property, refer to Debugging the Initial Stop State or Standby Mode. This property only appears when the selected microcontroller supports this facility.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Select [Yes] when debugging is to proceed in the initial stopped state and in standby mode.
		No	Select [No] when debugging is not to proceed in the initial stopped state and in standby mode.
Debug the GTM function	Select whether the GTM is to be debugged. This property only appears when the selected microcontroller supports this facility.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Select [Yes] when the GTM is to be debugged.
		No	Select [No] when the GTM is not to be debugged.
MCS to be debugged	Select the MCS on which to focus during debugging of the GTM. This property only appears when the selected microcontroller supports this facility.		
	Default	MCS0	
	Modifying	Select from the drop-down list.	
	Available values	A list of MCSs the device incorporates	
Debug the DFP function	Select whether the DFP is to be debugged. This property only appears when the selected microcontroller supports this facility.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Select [Yes] when the DFP is to be debugged.
		No	Select [No] when the DFP is not to be debugged.
IP address of the server	Specifies the IP address of the server to connect the OpenOCD.(IPv4) This property only appears when the selected microcontroller supports this facility and the [Debug the DFP function] property is set to [Yes].		
	Default	192.168.50.10	
	Modifying	**Directly enter from the keyboard.**	
	Available values	0.0.0.0 to 255.255.255.255	
Port number of the server	Specifies the port number of the server to connect the OpenOCD. This property only appears when the selected microcontroller supports this facility and the [Debug the DFP function] property is set to [Yes].		
	Default	9824	
	Modifying	**Directly enter from the keyboard.**	
	Available values	Integer number between 0 and 65535	

Use the DFP interlocking Resume / the DFP interlocking Halt function	Specifies whether to perform the DFP interlocking Resume / the DFP interlocking Halt when debugging DFP.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Use the DFP interlocking Resume / the DFP interlocking Halt function.
		No	Do not use the DFP interlocking Resume / the DFP interlocking Halt function.
Timeout time for waiting for the DFP execution request [s]	Specifies the timeout time waiting for the DFP execution request. This property only appears when the selected microcontroller supports this facility and the [Debug the DFP function] property is set to [Yes].		
	Default	10	
	Modifying	**Directly enter from the keyboard.**	
	Available values	Integer number between 0 and 65535	

(4) [Multi-core] [E1] [E20]

Debug target	Select wether to the debug target.	
	Default	[All cores mounted in the microcontroller]
	How to change	Specify with the Select Debug target dialog box. The Select debug target dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot change the setting on this panel).

(5) [External Trace] [IE850A]

The detailed information on external trace functions is displayed and its configuration can be changed.

Use external trace [IE850A]	Select whether to use the external trace.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Uses the external trace.
		No	Does not use the external trace.
Number of lanes of external trace [IE850A]	Select the number of lanes of the external trace. This property appears only when the [Use external trace] property is set [Yes]. If a number of lanes which differs from that of the connected target board has been specified when a debug tool is connected, the [Use external trace] property is changed to [No].		
	Default	- When the [Use emulation adapter] property is set to [Yes] 4 - When the [Use emulation adapter] property is set to [No] 1	
	Modifying	Select from the drop-down list. However, changeable only when disconnected from the debug tool	
	Available values	1, 2, 4	

Transfer speed of external trace [Gbps] [IE850A]	Select the transfer rate for the external trace. When [Auto] is selected, this will be the fastest value for the selected microcontroller.	
	Default	Auto
	Modifying	Select from the drop-down list.
	Available values	Auto, 1.250, 2.500, 3.125, 5.000, 6.250

(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] [Full-spec emulator][E1][E20]
The detailed information on the flash memory writing is displayed and its configuration can be changed.

Caution The properties in this category may vary with the selected microcontroller.
Properties in this category cannot be changed when CS+ is connected to the debug tool.

Security ID [E1][E20]	Specify the key code for ID authentication when connecting to the debug tool. There are microcontrollers that support ID authentication of 32 digits and microcontrollers that support ID authentication of 64 digits. When authentication by the key code set in this property fails, a connection error will occur.	
	Default	- For ID authentication of 32 digits FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF - For ID authentication of 64 digits FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFF
	Modifying	Directly enter from the keyboard.
	Available values	- For ID authentication of 32 digits 32 digits hexadecimal number (16 bytes) - For ID authentication of 64 digits 64 digits hexadecimal number (32 bytes)
Code Flash Access Password [E1][E20]	Displays the Access Password (hexadecimal value of 64 digits) when reading the code in the Code Flash. This property appears only when the selected microcontroller supports this function. When authentication by the Access Password set in this property fails, a connection error will occur.	
	Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFF
	Modifying	Directly enter from the keyboard.
	Available values	64 digits hexadecimal number (32 bytes)

Data Flash Access Password [E1][E20]	Displays the Access Password (hexadecimal value of 64 digits) when reading the code in the Data Flash. This property appears only when the selected microcontroller supports this function. When authentication by the Access Password set in this property fails, a connection error will occur.	
	Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFF
	Modifying	Directly enter from the keyboard.
	Available values	64 digits hexadecimal number (32 bytes)
OCD ID [IE850A][E2]	Specify the OCD ID (hexadecimal value of 64 digits) for connecting to the debug tool. This property appears only when the selected microcontroller supports this function. When authentication by the key code set in this property fails, a connection error will occur.	
	Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFF
	Modifying	Directly enter from the keyboard.
	Available values	64 digits hexadecimal number (32 bytes)
Customer ID [IE850A][E2]	Specify the ID code for the customer ID authentication. This property appears only when the selected microcontroller supports this function. When authentication by the key code set in this property fails, a connection error will occur.	
ID name (subproperty) [IE850A][E2]	Displays the ID name.	
	Default	Depends on the selected microcontroller
	Modifying	Changes not allowed
ID (subproperty) [IE850A][E2]	Specify the ID (64 digits in hexadecimal).	
	Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFF
	Modifying	Directly enter from the keyboard.
	Available values	64 digits hexadecimal number (32 bytes)
Data Flash ID [IE850A][E2]	Specify the ID code for the Data Flash ID authentication. This property appears only when the selected microcontroller supports this function. When authentication by the key code set in this property fails, a connection error will occur.	
ID name (subproperty) [IE850A][E2]	Displays the ID name.	
	Default	Depends on the selected microcontroller
	Modifying	Changes not allowed
ID (subproperty) [IE850A][E2]	Specify the ID (64 digits in hexadecimal).	
	Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFF
	Modifying	Directly enter from the keyboard.
	Available values	64 digits hexadecimal number (32 bytes)

Device specific ID [IE850A][E2]	Specify the device specific ID code for the ID authentication. This property appears only when the selected microcontroller supports this function. When authentication by the key code set in this property fails, a connection error will occur.		
ID name (subproperty) [IE850A][E2]	Displays the ID name.		
	Default	Depends on the selected microcontroller	
	Modifying	Changes not allowed	
Authenticate (subproperty) [IE850A][E2]	Select whether to authenticate this ID.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Authenticates this ID.
		No	Does not authenticate this ID.
ID (subproperty) [E2][IE850A]	Specify the ID (64 digits in hexadecimal).		
	Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFF	
	Modifying	Directly enter from the keyboard.	
	Available values	64 digits hexadecimal number (32 bytes)	
Set SVR parameters [E2][IE850A]	Specifies whether to set the SVR parameters. This property appears only when the selected microcontroller supports this function.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Specifies the SVR parameters from the debugging tool for use in starting the emulator.
		No	No SVR parameters are specified and the emulator is started with the values specified for the device.
SVR parameters [E2][IE850A]	Specify SVR parameters with sub-properties for the individual registers. For SVR parameters, all of SVRCFG0 to SVRCFG7 or SVRCFG0 to SVRCFG9 must be specified. If any of these parameters is blank, a connection error will occur. For details on the SVR parameters, refer to the user's manuals for the device and emulator you are using. This property is only displayed when the selected microcontroller supports this facility and [Yes] is selected for the [Specify SVR parameters] property.		
	Default	All of SVRCFG0 to SVRCFG7 or SVRCFG0 to SVRCFG9 are blank.	
	Modifying	Directly enter parameters for the individual registers from the keyboard.	
	Available values	8 digits hexadecimal number (4 bytes)	
SVRCFG*x* (subproperty) [E2][IE850A]	Specify SVR parameters.		
	Default	*Blank*	
	Modifying	Directly enter parameters for the individual registers from the keyboard.	
	Available values	8 digits hexadecimal number (4 bytes)	

<table>
<tr><td rowspan="6">Using the code flash self programming</td><td colspan="3">Select whether to rewrite the code flash by using the flash self library of the flash self programming function.
This property is not displayed for devices that do not have internal ROM.
Caution If the [Use software break] property in the [Debug Tool Setting] tab is set to [Yes], this property is fixed to [No] (changes not allowed).</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.
However, changeable only when disconnected from the debug tool</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Rewrites the code flash. If [Yes] is selected, the code flash will not be cached.</td></tr>
<tr><td>No</td><td>Does not rewrite the code flash.</td></tr>
<tr></tr>
<tr><td rowspan="5">Change the clock to flash writing</td><td colspan="3">Select whether to increase the clock speed temporarily for writing to the flash memory[Note].
This property is not displayed for devices that do not have internal ROM.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.
However, changeable only when disconnected from the debug tool</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Overclocks for writing to the flash memory.</td></tr>
<tr><td>No</td><td>Does not overclock for writing to the flash memory.</td></tr>
</table>

Note Selecting [Yes] may affect the peripheral system that is operating during a break because not only the CPU clock frequency but also the peripheral clock frequency changes.
When [No] is selected, the time of flash rewrite by the debugger operation will increase if the set clock speed is low.

(8) [Memory]
The detailed information on the memory is displayed and its configuration can be changed.

<table>
<tr><td rowspan="4">Map mode</td><td colspan="2">Select the map mode.
This property appears only when the selected microcontroller supports this function.
This property cannot be changed when CS+ is connected to the debug tool.
Caution [E2][IE850A]
If the microcontroller does not have the selected map mode, a connection error will occur.</td></tr>
<tr><td>Default</td><td>Single map mode</td></tr>
<tr><td>Modifying</td><td>Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>Single map mode, Double map mode, E2x-FCC2 compatible mode</td></tr>
<tr><td rowspan="4">Work RAM start address
[Full-spec emulator]
[E1][E20]</td><td colspan="2">Specify the first address of the working RAM area used by the debugger.
Specify the address as a 4-byte unit; if the input value is not a 4-byte unit, it is automatically adjusted.
The firmware of the debugger uses the range from the address where the working RAM is specified to start to the address corresponding to the size indicated in the [Work RAM size [Kbytes]] property.[Note 1]</td></tr>
<tr><td>Default</td><td>Blank is indicated immediately after a project has been created and the value depends on the selected microcontroller after connection to the emulator.</td></tr>
<tr><td>Modifying</td><td>Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td>An address within the Local RAM of the selected microcontroller or an address within the Retention RAM if there is no Local RAM[Note 2]</td></tr>
</table>

Work RAM size [Kbytes] [Full-spec emulator] [E1][E20]	Display the size of the working RAM area used by the debugger.	
	Default	Blank is indicated immediately after a project has been created and the value depends on the selected microcontroller after connection to the emulator.
	Modifying	Changes not allowed

Note 1. Since the contents of memory are saved and restored, this area can be used by the user program. However, the area allocated as the working RAM cannot be used in the following ways.

- As the source or destination for transfer by the DMA or DTS
- Use by external masters which are operating during a break

Note 2. For microcontrollers incorporating an Intelligent Cryptographic Unit Processor (ICUP), specify the LocalRAM area for the ICUP. For other microcontrollers, specify the LocalRAM area for CPU1.

(9) [Hardware-assisted Virtualization]
The detailed information on the hardware-assisted virtualization is displayed and its configuration can be changed.

Initial state of the hardware-assisted virtualization [Simulator]	Select the initial states of the virtualization facility with sub-properties for the individual CPUs.		
	Default	[Enable] for all CPUs	
	Modifying	Select the states from the drop-down lists for the individual CPUs.	
	Available values	Enable	The simulator is started with the virtualization facility enabled.
		Disable	The simulator is started with the virtualization facility disabled.

(10) [Configuration] [Simulator]
The property in this category is always disabled.

(11) [CPU Virtualization Support Function]
The property in this category is always disabled. Do not change it.

[Debug Tool Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.

(1) [Memory]
(2) [Access Memory While Running]
(3) [Set Event While Running] [Full-spec emulator][E1][E20]
(4) [Reset While Running] [Full-spec emulator][E1][E20]
(5) [E2 Expansion Interface] [E2]
(6) [Break] [Full-spec emulator][E1][E20]
(7) [Trace]
(8) [Trace Data]
(9) [Trace Memory]
(10) [Trace Event][Full-spec emulator][E1][E20]
(11) [Trace Time Tag] [Simulator]
(12) [Output Software Trace from LPD] [E2]
(13) [Timer]
(14) [Mask for Input Signal] [Full-spec emulator][E1][E20]
(15) [Coverage] [Simulator]
(16) [Simulator GUI] [Simulator]
(17) [Multi core] [Full-spec emulator][E1][E20]
(18) [Hardware-assisted Virtualization]
(19) [Step function]

[Description of each category]

(1) [Memory]
The detailed information on memories is displayed and its configuration can be changed.

Memory mappings	The state of memory mapping is displayed for each type of memory area[Note 1]. **Caution 1.** The external memory area can be accessed only when a target board mounted with memory (e.g., RAM) other than flash memory is used for the external memory area. **Caution 2.** The memory mapping cannot be added/deleted. [Full-spec emulator][E1][E20] Remark When the selected microcontroller supports multi-core, this property displays the memory mapping status regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").	
	Default	[*Sum total by microcontroller's inherent type of memory mapped area*]
	Modifying	Specify with the Memory Mapping dialog box. The Memory Mapping dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot change the memory mapping on this panel).
	Displayed Content	Displays the memory mapping status for each type of memory area. The following detailed information is displayed by clicking the "+" mark of each memory type. - Memory type - Start address - End address - Access width[bits][Note 2] [Full-spec emulator][E1][E20]

Verify on writing to memory [Full-spec emulator] [E1][E20]	Select whether to perform a verify check when the memory value is initialized. Select [Yes] to perform verification after download or when values are changed in the Memory panel/Watch panel (default).		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Executes the verify check.
		No	Does not execute the verify check.
Permit writing to the writing prohibited area [Simulator]	Select whether a break is to be generated in case of writing to a writing prohibited area. When [No] is selected, a break occurs in response to writing to a writing prohibited area. When [Yes] is selected, writing to a writing prohibited area proceeds without a break.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	A break does not occur.
		No	A break occurs.
Select I/O modules which details are displayed	Select whether to select I/O modules which details are displayed. When [Yes] is selected, details on the I/O modules selected in the [I/O modules] property are displayed. This property only appears when the selected microcontroller requires it.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Details on the I/O modules selected in the [I/O modules] property are displayed.
		No	Details on all I/O modules are displayed.
I/O modules	Select the I/O modules which details are displayed.		
	Default	[*Total number of I/O modules*]	
	Modifying	Select in the Select I/O Modules dialog box The Select I/O Modules dialog box is opened by clicking the [...] button that appears on the right edge of the field when you select this property (you can not select the I/O modules on this panel).	

Note 1. The type is of the memory mapping area registered in the device file.

Note 2. This appears when the memory type is External Memory, Serial Flash and SD RAM.

(2) [Access Memory While Running]
The detailed information on memory accesses while executing a program (real-time display update function: see "2.12.1.4 Display/modify the memory contents during program execution") is displayed and its configuration can be changed.

Access during the execution **[Full-spec emulator]** **[E1][E20]**	Select whether to allow access to the internal RAM area during execution of a program.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Accesses to the internal RAM area during execution of a program.
		No	Does not access to the internal RAM area during execution of a program.

Update display during the execution	Select whether to update the display in the Memory panel/Watch panel during a program execution.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Updates the display during program execution.
		No	Does not update the display during program execution.
Display update interval[ms]	Specify the interval in 100ms unit to update the contents in the Memory panel/Watch panel display while executing a program. This property appears only when the [Update display during the execution] property is set to [Yes].		
	Default	500	
	Modifying	Directly enter from the keyboard.	
	Available values	Integer number between 100 and 65500 (rounding up the fractions less than 100 ms)	

(3) [Set Event While Running] [Full-spec emulator][E1][E20]
The detailed information on the function of the event setting during program execution is displayed and its configuration can be changed.

Set event by stopping execution momentarily	Select whether to forcibly pause the execution for events that cannot be set while executing the program or operating the tracer/timer. For details on the event types that are affected by this property, see "2.19.7.2 Event types that can be set and deleted during execution".		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Sets these events by stopping the program execution or the tracer/timer operation momentarily.
		No	Does not allow to set these events during program execution or the tracer/timer operation.

(4) [Reset While Running] [Full-spec emulator][E1][E20]
The detailed information on the reset operation during program execution is displayed and its configuration can be changed.

Use the force reset	Select whether to apply a forced reset when a reset or forced break fails during execution of the user program. When [Yes] is selected and a reset fails for either of the following reasons, a forced reset is automatically applied. - When the clock supply is stopped, etc., so that forced breaks are disabled - When a core (PE) is in the initially stopped state After a forced reset succeeds, all cores (PE) enter the break state after the reset.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Applies a forced reset
		No	Does not apply a forced reset

(5) [E2 Expansion Interface] [E2]
The detailed information on the E2 expansion interface is displayed and its configuration can be changed.

<table>
<tr><td>External trigger input</td><td colspan="3">Make settings for the external trigger input.
You can select different actions for each channel.</td></tr>
<tr><td rowspan="3">Channel number
(Subproperty)</td><td colspan="3">The channel number is displayed.</td></tr>
<tr><td>Default</td><td colspan="2">0 or 1</td></tr>
<tr><td>Modifying</td><td colspan="2">Changes not allowed</td></tr>
<tr><td rowspan="5">Use
(Subproperty)</td><td colspan="3">Specify whether to use the external trigger input for this channel number.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Uses the external trigger signal input through the selected channel.</td></tr>
<tr><td>No</td><td>Does not use the external trigger signal input through the selected channel.</td></tr>
<tr><td rowspan="8">Input signal
(Subproperty)</td><td colspan="3">Specify the input signal.</td></tr>
<tr><td>Default</td><td colspan="2">Rising Edge</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="5">Available values</td><td>Rising Edge</td><td>Detects rising edges of the external trigger signal input through the selected channel.</td></tr>
<tr><td>Falling Edge</td><td>Detects falling edges of the external trigger signal input through the selected channel.</td></tr>
<tr><td>Both Edges</td><td>Detects both rising and falling edges of the external trigger signal input through the selected channel.</td></tr>
<tr><td>High</td><td>Detects the high level of the external trigger signal input through the selected channel.</td></tr>
<tr><td>Low</td><td>Detects the low level of the external trigger signal input through the selected channel.</td></tr>
<tr><td rowspan="5">Action when inputting the external trigger
(Subproperty)</td><td colspan="3">Select the action that the E2 emulator will take in response to input of the external trigger signal.</td></tr>
<tr><td>Default</td><td colspan="2">Stop trace</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Stop trace</td><td>Stops the recording of trace data in response to detection of the external trigger signal input through the selected channel.</td></tr>
<tr><td>Stop program</td><td>Stops execution of the program and the recording of trace data in response to detection of the external trigger signal input through the selected channel.</td></tr>
<tr><td>External trigger output</td><td colspan="3">Set the settings related to the external trigger output.
You can select different actions for each channel.</td></tr>
<tr><td rowspan="3">Channel number
(Subproperty)</td><td colspan="3">The channel number is displayed.</td></tr>
<tr><td>Default</td><td colspan="2">0 or 1</td></tr>
<tr><td>Modifying</td><td colspan="2">Changes not allowed</td></tr>
</table>

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	

(6) [Break] [Full-spec emulator][E1][E20]
The detailed information on break functions is displayed and its configuration can be changed.
See "2.11 Stop Programs (Break)" for details on the break function and this category configuration.

Use software break	Select whether to use the Software break function [Full-spec emulator][E1][E20][Note].		
	Default	No	
	Modifying	Select from the drop-down list. Note that changes can be made only when program execution is halted.	
	Available values	Yes	Uses the software break function.
		No	Does not use the software break function.
First using type of breakpoint	Select the type of the breakpoint to use with priority when setting it at the source line or the execution address with a one click operation of the mouse in the Editor panel/Disassemble panel.		
	Default	Software break	
	Modifying	Select from the drop-down list.	
	Available values	Software break	Sets software breakpoint with priority.
		Hardware break	Sets hardware breakpoint with priority.
Stop emulation of peripherals when stopping	Select whether to terminate the peripheral emulation while stopping the program execution (Peripheral Break).		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Terminates the peripheral emulation.
		No	Does not terminate the peripheral emulation.

Note If this property is set to [No] after you have used the software break function, all software break events and Printf events that have been set will be disabled. Selecting [Yes] in this state does not automatically restore the events, so you will need to manually enable them.

(7) [Trace]

The detailed information on trace functions is displayed and its configuration can be changed (see "2.14.1 Configure the trace operation").

Caution 1. [Full-spec emulator][E1][E20]
Properties in this category cannot be changed during program execution.

Caution 2. [E1][E20]
If the trace function is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (the trace function cannot be used).

Trace priority [Full-spec emulator] [E1][E20]	Select which item should be given priority when collecting the trace data[Note 1].		
	Default	Speed priority	
	Modifying	Select from the drop-down list.	
	Available values	Speed priority	Traces giving priority to the real-time performance.
		Data priority	Traces after stopping the execution pipeline of the CPU temporarily so that no data is missed.
Use trace function [Simulator]	Select whether to use the trace function[Note 2].		
	Default	No	
	Modifying	Select from the drop-down list. Note that changes can be made only when program execution is halted.	
	Available values	Yes	Uses trace functions.
		No	Does not use trace functions.
Enable trace data complement [Full-spec emulator]	Select whether to enable complement display when displaying the collected trace data. By enabling complement display, instructions between branch instructions that cannot be traced by hardware can be displayed. This setting will be applied from the next acquisition of trace data.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Performs complementary display of trace data.
		No	Does not perform complementary display of trace data.
Trace target	Select the core to be traced[Note 3]. When the core name selected for this property is changed, also change the value of the [Timer event target] property in the [Timer] category.		
	Default	Debug core only	
	Modifying	Select from the drop-down list.	
	Available values	Debug core only	Collects the trace data regarding the currently selected PE (default).
		All core	Collects the trace data for all PEs.
		Core name [Full-spec emulator][E1][E20]	Collects the trace data for the selected *core name*.

Use SPID filter [IE850A][E2]	Select whether to use the SPID filter. When the SPID filter is in use, only information on tracing which belongs to the SPIDs specified with the [SPID filter] property is collected. This property appears only when the selected microcontroller supports this function.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Uses the SPID fillter.
		No	Does not use the SPID fillter.
SPID filter [IE850A][E2]	Select the SPID for which trace data are to be output. This property appears only when the [Use SPID filter] property is set to [Yes].		
	Default	[32]	
	Modifying	Specify with the Select SPID filter dialog box. The Select SPID filter dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot change the setting on this panel).	
	Displayed Content	The total number of SPID for which trace data are to be output	

Note 1. **[Full-spec emulator][E1][E20]**
The trace memory is cleared when you change the setting of this property.

Note 2. This property is automatically set to [Yes] when selecting [Start Tracing]/[Stop Tracing] from the context menu in the Editor panel/Disassemble panel.

Note 3. **[Simulator]**
The trace memory is cleared when you change the setting of this property.

(8) [Trace Data]

Caution 1. [Full-spec emulator][E1][E20]
Properties in this category cannot be changed during program execution.

Caution 2. [E1][E20]
If the trace function is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (the trace function cannot be used).

Trace the branch PC and the data access [Simulator]	Select whether to collect PC values for the source and destination instructions of branching during program execution and the PC values and information on the data for instructions leading to access-related events that occur during program execution as trace data[Note].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Collects PC values and information as trace data.
		No	Does not collect PC values and information as trace data.
Trace the branch PC [Full-spec emulator] [E1][E20]	Select whether to collect PC values for source/destination instructions of branching during program execution as trace data.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Collects PC values as trace data.
		No	Does not collect PC values as trace data.

<table>
<tr><td rowspan="5">Trace the data access
[Full-spec emulator][E1][E20]</td><td colspan="3">Select whether to collect data information on access-related events that occurred during program execution as trace data.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects data information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect data information as trace data.</td></tr>
<tr><td rowspan="5">Trace the fetch address of the data access
[Full-spec emulator][E1][E20]</td><td colspan="3">Select whether to collect PC values for instructions of access-related events that occurred during program execution as trace data.
When PC values are collected, the executed instructions are displayed in the Trace panel.
This property appears only when the [Trace the data access] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects PC values as trace data.</td></tr>
<tr><td>No</td><td>Does not collect PC values as trace data.</td></tr>
<tr><td rowspan="5">Trace local variable access
[Full-spec emulator][E1][E20]</td><td colspan="3">Select whether to collect data information on access-related events for accesses to local variables that occurred during program execution as trace data.
This property appears only when the [Trace the data access] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects data information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect data information as trace data.</td></tr>
<tr><td rowspan="5">Trace the transition information of CPU operation mode
[IE850A][E2]</td><td colspan="3">Select whether to record transitions of the CPU operating mode that occur during the execution of the program as trace data. When transitions of the CPU operating mode are being recorded, the CPU operating mode is displayed for each item of trace data in the Trace panel.
This property appears only when the selected microcontroller supports this function.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects data information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect data information as trace data.</td></tr>
<tr><td rowspan="5">Trace the software trace</td><td colspan="3">Select whether to collect information on trace output instructions to be embedded that were generated during program execution as trace data[Note].</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect information as trace data.</td></tr>
</table>

<table>
<tr><td rowspan="4">Trace the DBCP</td><td colspan="3">Select whether to collect information on DBCP that were generated during program execution as trace data[Note].
This property appears only when the [Trace the software trace] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>Yes
No</td><td>Collects information as trace data.
Does not collect information as trace data.</td></tr>
<tr><td rowspan="4">Trace the DBTAG</td><td colspan="3">Select whether to collect information on DBTAG that were generated during program execution as trace data[Note].
This property appears only when the [Trace the software trace] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>Yes
No</td><td>Collects information as trace data.
Does not collect information as trace data.</td></tr>
<tr><td rowspan="4">Trace the fetch address of the DBTAG
[Full-spec emulator][E1][E20]</td><td colspan="3">Select whether to collect information on DBTAG that were generated during program execution, along with the values of addresses where the DBTAG instructions were executed.
This property appears only when the [Trace the software trace] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>Yes
No</td><td>Collects information as trace data.
Does not collect information as trace data.</td></tr>
<tr><td rowspan="4">Trace the DBPUSH</td><td colspan="3">Select whether to collect information on DBPUSH that were generated during program execution as trace data[Note].
This property appears only when the [Trace the software trace] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>Yes
No</td><td>Collects information as trace data.
Does not collect information as trace data.</td></tr>
<tr><td rowspan="4">Trace the fetch address of the DBPUSH
[Full-spec emulator][E1][E20]</td><td colspan="3">Select whether to collect information on DBPUSH that were generated during program execution, along with the values of addresses where the DBTAG instructions were executed.
This property appears only when the [Trace the software trace] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>Yes
No</td><td>Collects information as trace data.
Does not collect information as trace data.</td></tr>
</table>

<table>
<tr><td rowspan="5">Trace the GTM</td><td colspan="3">Select whether to collect the results of tracing of the GTM.
This property appears only when the [Debug the GTM function] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect information as trace data.</td></tr>
<tr><td rowspan="4">Channel to get the trace of the MCS</td><td colspan="3">Select the MCS channel to get the trace.
This property appears only when the [Trace the GTM] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">All</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td colspan="2">All,Ch0,Ch1,Ch,Ch3,Ch4,Ch5,Ch6,Ch7</td></tr>
<tr><td rowspan="5">Trace the branch PC of the MCS</td><td colspan="3">Select whether to collect PC values for the source and destination instructions in branch processing that occurred during GTM execution as trace data.
This property appears only when the [Trace the GTM] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect information as trace data.</td></tr>
<tr><td rowspan="5">Trace the data access of the MCS</td><td colspan="3">Select whether to collect data information on access-related events that occurred during GTM execution as trace data.
This property appears only when the [Trace the GTM]property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Collects information as trace data.</td></tr>
<tr><td>No</td><td>Does not collect information as trace data.</td></tr>
</table>

Note **[Simulator]**
The trace memory is cleared when you change the setting of this property.

(9) [Trace Memory]

Caution 1. [Full-spec emulator][E1][E20]
Properties in this category cannot be changed during program execution.

Caution 2. [E1][E20]
If the trace function is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (the trace function cannot be used).

<table>
<tr><td rowspan="5">Trace memory</td><td colspan="3">Select the area to store the trace data with the internal trace function.
If the device used for debugging cannot output the internal trace to the Global Emulation RAM when connecting with [Global Emulation RAM] selected, it is automatically changed to [Internal trace memory].
[IE850A]
This property appears only when the selected the [Debug mode] property in the [External Trace] category in [Connect Settings] tab is set to [Use external trace].</td></tr>
<tr><td>Default</td><td colspan="2">Internal trace memory</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td colspan="2">Internal trace memory</td></tr>
<tr><td colspan="2">Global Emulation RAM</td></tr>
<tr><td rowspan="4">Start address of the trace memory</td><td colspan="3">Specifies the start address of the area where the trace data is stored when the trace data is stored in the Global Emulation RAM by the internal trace.
If a setting error occurs during connection, set the start address of the Global Emulation RAM as the initial value.</td></tr>
<tr><td>Default</td><td colspan="2">Blank</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">An address within the Global Emulation RAM of the selected microcontroller.</td></tr>
<tr><td rowspan="4">Trace memory size[frames]
[IE850A]
[Full-spec emulator]</td><td colspan="3">Select the memory size for storing the trace data by the trace frame numbers[Note 1, 2, 3].</td></tr>
<tr><td>Default</td><td colspan="2">[IE850A][Full-spec emulator]
8K
[Simulator]
4K</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td colspan="2">[IE850A]
2K, 4Kt, 8K, 16K, 32K, 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M, 16M, 32M, 64M, 128M, 256M, 512M, 1G
[Full-spec emulator]
8K, 16K, 32K, 64K, 128K, 256K, 512K
[Simulator]
4K, 8K, 12K, 16K, 20K, 24K, 28K, 32K, 36K, 40K, 44K, 48K, 52K, 56K, 60K, 64K, 128K 192K, 256K, 320K, 384K, 448K, 512K, 576K, 640K, 704K, 768K, 832K, 896K, 960K, 1M, 2M, 3M</td></tr>
<tr><td rowspan="5">Clear trace memory before running</td><td colspan="3">Select whether to clear the trace memory before executing.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Clears the trace memory.</td></tr>
<tr><td>No</td><td>Does not clear the trace memory.</td></tr>
</table>

Operation after trace memory is full	Select the operation after the trace memory is full with the collected trace data[Note 2].		
	Default	Non stop and overwrite to trace memory	
	Modifying	Select from the drop-down list.	
	Available values	Non stop and over-write to trace memory	Continues overwriting trace data even after trace memory is used up.
		Stop trace[Note 4]	Stops overwriting trace data when trace memory is used up (the program execution will not be stopped).
		Stop	Stops running the program and overwriting trace data when trace memory is used up.

Note 1. The trace frame is a unit of trace data. Each fetch/write/read uses one trace frame.

Note 2. **[Full-spec emulator][E1][E20]**
The trace memory is cleared when you change the setting of this property.

Note 3. **[Simulator]**
The trace memory is cleared when you change the setting of this property.

Note 4. **[E1][E20]**
This item is not displayed when the [Trace priority] property is set to [Data priority].
[IE850A]
This item is not displayed when the [Use external trace] property on the [Connect Settings] tab is set to [No] and the [Trace priority] property is set to [Data priority].

(10) [Trace Event][Full-spec emulator][E1][E20]

Caution 1. [Full-spec emulator][E1][E20]
Properties in this category cannot be changed during program execution.

Caution 2. [E1][E20]
If the trace function is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (the trace function cannot be used).

Trace range setting [Full-spec emulator] [E1][E20]	Select the range of trace data to be collected.		
	Default	Traces section	
	Modifying	Select from the drop-down list.	
	Available values	Traces section	Collects the execution history as trace data within the section specified with a trace start event and a trace end event.
		Traces out of range	Collects the execution history as trace data outside the range specified with a trace start event and a trace end event.

(11) [Trace Time Tag] **[Simulator]**
The detailed information on Trace Time Tagis displayed and its configuration can be changed.

Accumulate trace time [Simulator]	Select whether to display the accumulated tracing time in the Trace panel[Note].		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Displays the accumulated tracing time.
		No	Displays the trace time with differential value.

<table>
<tr><td rowspan="4">Rate of frequency division of trace time tag
[Simulator]</td><td colspan="2">Select the frequency division ratio of the counter to be used for time tag display (the [Time] item in the Trace panel)Note.
Changing the frequency division ratio here changes the number of clocks necessary to count up a counter value which is displayed in the time tag of the branch PC and the data access.</td></tr>
<tr><td>Default</td><td>1/1</td></tr>
<tr><td>Modifying</td><td>Select from the drop-down list.</td></tr>
<tr><td>Available values</td><td>1/1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256, 1/512, 1/1K, 1/4K, 1/8K, 1/16K, 1/64K, 1/256K, 1/1M</td></tr>
</table>

Note The trace memory is cleared when you change the setting of this property.

(12) [Output Software Trace from LPD] [E2]
The detailed information on software tracing through LPD output is displayed and its configuration can be changed (see "2.14.1 Configure the trace operation").

Caution 1. [E2]
Properties in this category cannot be changed during program execution.

Caution 2. [E2]
If software tracing through LPD output is not mounted on the microcontroller used, all properties in this category become unchangeable after connecting to the debug tool (software tracing through LPD output cannot be used).

<table>
<tr><td rowspan="5">Output the software trace from the LPD</td><td colspan="3">Select whether to output the software trace from the LPD.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Outputs the software trace from the LPD.</td></tr>
<tr><td>No</td><td>Does not output the software trace from the LPD.</td></tr>
<tr><td rowspan="4">Target when outputting the software trace from the LPD</td><td colspan="3">Select the target when outputting the software trace from the LPD.
This property appears only when the [Output the software trace from the LPD] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Core name</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.
However, changeable only when disconnected from the debug tool.</td></tr>
<tr><td>Available values</td><td>Core name</td><td>Outputs trace data from the LPD for the selected core name.</td></tr>
<tr><td rowspan="5">Output the DBCP</td><td colspan="3">Select whether to output information on DBCP that were generated during program execution as trace data from the LPD.
This property appears only when the [Output the software trace from the LPD] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Outputs information from the LPD.</td></tr>
<tr><td>No</td><td>Does not output information from the LPD.</td></tr>
</table>

Output the DBTAG	Select whether to output information on DBTAG that were generated during program execution as trace data from the LPD. This property appears only when the [Output the software trace from the LPD] property is set to [Yes].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Outputs information from the LPD.
		No	Does not output information from the LPD.
Output the fetch address of the DBTAG	Select whether to output information on DBTAG that were generated during program execution as trace data from the LPD, along with the values of addresses where the DBTAG instructions were executed. This property appears only when the [Output the software trace from the LPD] property is set to [Yes].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Collects information as trace data.
		No	Does not collect information as trace data.
Output the DBPUSH	Select whether to output information on DBPUSH that were generated during program execution as trace data from the LPD. This property appears only when the [Output the software trace from the LPD] property is set to [Yes].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Collects information as trace data.
		No	Does not collect information as trace data.
Output the fetch address of the DBPUSH	Select whether to output information on DBPUSH that were generated during program execution as trace data from the LPD, along with the values of addresses where the DBPUSH instructions were executed. This property appears only when the [Output the software trace from the LPD] property is set to [Yes].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Collects information as trace data.
		No	Does not collect information as trace data.
Priority when outputting the software trace data from the LPD	Select which item should be given priority when outputting the software trace from the LPD. This property appears only when the [Output the software trace from the LPD] property is set to [Yes].		
	Default	Speed priority	
	Modifying	Select from the drop-down list.	
	Available values	Speed priority	Outputs the software trace from the LPD giving priority to the real-time performance.
		Data priority	Outputs the software trace from the LPD after stopping the execution pipeline of the CPU temporarily so that no data is missed.

Operation after the recording memory is full	Select the operation after the recording memory is full with software trace data. This property appears only when the [Output the software trace from the LPD] property is set to [Yes].		
	Default	Overwrite the record memory and continue	
	Modifying	Select from the drop-down list.	
	Available values	Overwrite the record memory and continue	Continues overwriting older software trace data even after the recording memory is used up.
		Stop recording	Stops outputting software trace data when the recording memory is used up (the program execution will not be stopped).
		Stop program	Stops running the program and outputting software trace data when the recording memory is used up.

(13) [Timer]
The detailed information on timer functions is displayed and its configuration can be changed.

Use timer function [Simulator]	Select whether to use the timer function.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Uses timer functions.
		No	Does not use timer functions.
Timer event target [Full-spec emulator][E1][E20]	Select the timer event target. When the core name selected for this property is changed, also change the value of the [Trace target] property in the [Trace] category. This property appears only when the selected microcontroller supports this function and the [Debug mode] property in the [Multi core] category is set to [Async debug mode].		
	Default	*Core name*	
	Modifying	Select from the drop-down list.	
	Available values	*Core name*	Enables the specification of a timer event for the selected core.

(14) [Mask for Input Signal] [Full-spec emulator][E1][E20]
The detailed information on the masking input signal is displayed and its configuration can be changed.

Mask WAIT signal	Select whether to mask WAIT signal to prevent the signal input to emulators.		
	Default	[Full-spec emulator] No [E1][E20] Yes	
	Modifying	Select from the drop-down list[Note 1].	
	Available values	Yes	Masks WAIT signal.
		No	Does not mask WAIT signal.

<table>
<tr><td rowspan="5">Mask RESET signal</td><td colspan="3">Select whether to mask RESET signal to prevent the signal input to emulators.</td></tr>
<tr><td>Default</td><td colspan="2">[Full-spec emulator]
No
[E1][E20]
Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list[Note 1.]</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Masks RESET signal.</td></tr>
<tr><td>No</td><td>Does not mask RESET signal.</td></tr>
<tr><td rowspan="5">Select the RESET signal to mask</td><td colspan="3">Select a RESET signal to be masked.
This property appears only when the [Mask RESET signal] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">[Full-spec emulator]
TARGET RESET signal
[E1][E20]
TARGET RESET signal and INTERNAL RESET signal</td></tr>
<tr><td>Modifying</td><td colspan="2">[Full-spec emulator]
Select from the drop-down list.
[E1][E20]
Changes not allowed</td></tr>
<tr><td rowspan="2">Available values-Note 2</td><td>TARGET RESET signal</td><td>Masks TARGET RESET signal.</td></tr>
<tr><td>TARGET RESET signal and INTERNAL RESET signal</td><td>Masks TARGET RESET signal and INTERNAL RESET signal.</td></tr>
<tr><td rowspan="5">Mask PWRGD signal
[Full-spec emulator]</td><td colspan="3">Select whether to mask PWRGD signal to prevent the signal input to emulators.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list[Note 1, 3.]</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Masks PWRGD signal.</td></tr>
<tr><td>No</td><td>Does not mask PWRGD signal.</td></tr>
</table>

Note 1. **[Full-spec emulator]**
When the [Connecting with target board] property in the [Connection with Target Board] [Full-spec emulator][E1][E20] category on the [Connect Settings] tab is set to [No], this property is fixed to [Yes] automatically after connecting to the debug tool (changes not allowed).

Note 2. **[Full-spec emulator]**
If this property cannot be set as [TARGET RESET signal] in the POD, it is fixed to [TARGET RESET signal and INTERNAL RESET signal] after connection to the debug tool (changing from this setting is not allowed).

Note 3. If this property cannot be set as [Yes] in the POD, it is automatically fixed to [No] after connection to the debug tool (changing from this setting is not allowed).

(15) [Coverage] [Simulator]
The detailed information on coverage functions is displayed and its configuration can be changed.

<table>
<tr><td rowspan="5">Use coverage function</td><td colspan="3">Select whether to use the coverage function.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Uses coverage functions</td></tr>
<tr><td>No</td><td>Does not use coverage functions</td></tr>
</table>

Reuse coverage result	Select whether to load/save the coverage measurement result when connecting to or disconnecting from the debug tool. This property appears only when the [Use coverage function] property is set to [Yes].		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Loads/saves the coverage measurement result.
		No	Does not load/save the coverage measurement result.
Coverage area of measure-ment(1MBytes)	Specify the area that performs coverage measurement. Specify the start address of any 1 Mbyte space other than the internal ROM area. This property appears only when the [Use coverage function] property is set to [Yes].		
	Default	100000	
	Modifying	Directly enter from the keyboard.	
	Available values	Address without the address range of the internal ROM area (symbols cannot be used).	

(16) [Simulator GUI] [Simulator]
The detailed information on the Simulator GUI function is displayed and its configuration can be changed.

Caution If a microcontroller whose Simulator does not support peripheral function simulations is selected (i.e. the selected microcontroller supports only a instruction simulator), all properties in this category become invalid.

Display Simulator GUI	Select whether to display the Simulator GUI window to use the Simulator GUI function.		
	Default	Yes	
	Modifying	Select from the drop-down list. Note that changes can be made only when program execution is halted.	
	Available values	Yes	Displays the Simulator GUI window.
		No	Does not displays the Simulator GUI window.
Display Simulator GUI on top of other windows	Select whether to display the Simulator GUI window in the forefront when program execution starts. This property appears only when the [Display Simulator GUI] property is set to [Yes].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Displays it in the forefront.
		No	Does not display it in the forefront.

(17) [Multi core] [Full-spec emulator][E1][E20]
The detailed information on the control method of a multi-core is displayed and its configuration can be changed.

<table>
<tr><td rowspan="5">Debug mode</td><td colspan="3">Select the control method of a multi-core.
Caution The cores that can collect trace data differ depending on the selection of this property.
For the selection of the cores to collect trace data, see "2.14.1 Configure the trace operation".
See "2.9 Select a Core (PE)" for selecting the debug target.
This property appears only when the selected microcontroller is a multi-core.
This property can be changed only while all cores are stopped.</td></tr>
<tr><td>Default</td><td colspan="2">Sync debug mode</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Sync debug mode</td><td>Synchronizes execution and stop of all cores mounted in the microcontroller.
For the cores that can collect trace data, [Debug core only] or [All core] can be selected by the [Trace target] property in the [Trace] category.</td></tr>
<tr><td>Async debug mode</td><td>Controls execution and stop of only the core that is selected to be debugged.
The core that can collect trace data is only the core selected by the [Trace target] property in the [Trace] category.</td></tr>
<tr><td rowspan="5">Debug initial stop state</td><td colspan="3">Select whether to debug the initial stop state of the CPU.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>When the program is restarted, the CPU enters the initial stop state (which is the state the CPU enters on release from the reset state).</td></tr>
<tr><td>No</td><td>When the program is restarted, the CPU does not enter the initial stop state (which is the state the CPU enters on release from the reset state).
The CPU enters the break state before the program starts running.</td></tr>
</table>

(18) [Hardware-assisted Virtualization]
The detailed information on the control method of step execution is displayed and its configuration can be changed.

<table>
<tr><td rowspan="5">Select contexts on debug target</td><td colspan="3">Specify whether to select to the contexts on the debug target.
When [Yes] is selected, only the contexts selected in the [Contexts on debug target] property are handled as the targets for debugging.
This property appears only when the selected microcontroller supports this function.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>The contexts for debugging are to be specified.</td></tr>
<tr><td>No</td><td>The contexts for debugging are not to be specified.</td></tr>
</table>

<table>
<tr><td rowspan="4">Contexts on debug target</td><td colspan="3">Select wether to the contexts on the debug target.
Select the contexts for debugging per core of the selected microcontroller.
This property appears only when the [Select contexts on debug target] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">[The total number of cores in the selected microcontroller]</td></tr>
<tr><td>Modifying</td><td colspan="2">Specify with the Select Contexts on Debug target dialog box.
The Select Contexts on Debug target dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot change the setting on this panel).</td></tr>
<tr><td>Displayed Content</td><td colspan="2">Displays the selected state of the contexts for debugging.</td></tr>
<tr><td rowspan="5">Skip contexts other than debug targets</td><td colspan="3">Select whether to skip the contexts that are not for debugging.
When [Yes] is selected, if a break occurs in a context that has not been selected in the [Contexts on debug target] property, program execution is automatically continued until a transition to a context for debugging, and the break occurs on completion of this transition.
This property appears only when the [Select contexts on debug target] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Contexts that are not for debugging are skipped.</td></tr>
<tr><td>No</td><td>Contexts that are not for debugging are not skipped.</td></tr>
</table>

(19) [Step function]
The detailed information on the control method of step execution is displayed and its configuration can be changed.

<table>
<tr><td rowspan="5">Skip target section</td><td colspan="3">Select whether to skip the target section.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Skips the target section.</td></tr>
<tr><td>No</td><td>Does not skip the target section.</td></tr>
<tr><td rowspan="3">Target section</td><td colspan="3">Specify the target section.
This property appears only when the [Skip target section] property is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">[Number of sections to skip]</td></tr>
<tr><td>Modifying</td><td colspan="2">Specify with the Specified Section dialog box.
The Specified Section dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (The sections to be skipped cannot be specified from this panel.).</td></tr>
</table>

[Download File Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.
For details on the download function, see "2.6 Download/Upload Programs".

(1) [Download]
(2) [Debug Information]

[Description of each category]

(1) [Download]
The detailed information on download is displayed and its configuration can be changed.

Download files	Specify the file to download[Note 1]. The names of files to be downloaded and the download conditions are listed in the lower area.		
	Default	*[Number of files to download]*	
	Modifying	Specify with the Download Files dialog box. The Download Files dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify the file to download on this panel).	
CPU Reset after download	Select whether to reset the CPU after downloading.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Resets the CPU after downloading.
		No	Does not reset the CPU after downloading.
Erase flash ROM before download [Full-spec emulator][E1][E20]	Select whether to erase the flash ROM before downloading.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Erases the flash ROM before downloading.
		No	Does not erase the flash ROM before downloading.
Automatic change method of event setting position	Select how to perform the setting again if the file is downloaded again, and the location (address) set for the currently set event changes to midway in the instruction[Note 2].		
	Default	Suspend event	
	Modifying	Select from the drop-down list.	
	Available values	Move to the head of instruction	Sets the event to the top address of the instruction.
		Suspend event	Disables the event (suspended state).
Allow downloading to Configuration Setting Area [IE850A][E2]	Select whether to allow downloading to Configuration Setting Area. Reconnect to the debug tools when after downloading to Configuration Setting Area.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloading to Configuration Setting Area is allowed.
		No	Downloading to Configuration Setting Area is not allowed.

Allow downloading to Block Protection Area [IE850A][E2]	Select whether to allow downloading to Block Protection Area. Reconnect to the debug tools when after downloading to Block Protection Area.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloading to Block Protection Area is allowed.
		No	Downloading to Block Protection Area is not allowed.
Allow downloading to Security Setting Area [IE850A][E2]	Select whether to allow downloading to Security Setting Area. Reconnect to the debug tools when after downloading to Security Setting Area.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloading to Security Setting Area is allowed.
		No	Downloading to Security Setting Area is not allowed.
Allow downloading to Switch Area [IE850A][E2]	Select whether to allow downloading to Switch Area. Reconnect to the debug tools when after downloading to Switch Area.		
	Default	No	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloading to Switch Area is allowed.
		No	Downloading to Switch Area is not allowed.

Note 1. Files specified as build targets in a main project or sub-project cannot be deleted from the target files to download (These files are automatically registered as download files by default).
See "Table 2.4 Downloadable File Formats" for downloadable file format.

Note 2. This property setting works only for the location setting of events without the debug information. The location setting of events with the debug information is always moved to the beginning of the source text line.

(2) [Debug Information]
The detailed information on debugging is displayed and its configuration can be changed.

Execute to the specified symbol after CPU Reset	Select whether to execute the program to the specified symbol position after CPU reset.		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Executes the program to the specified symbol position after CPU reset.
		No	Does not execute the program after CPU reset.
Specified symbol	Specify the position at which the program is stop after CPU reset. This property appears only when the [Execute to the specified symbol after CPU Reset] property is set to [Yes].		
	Default	_main	
	Modifying	Directly enter from the keyboard.	
	Available values	Address expression between 0 and the *"end address of the address space"*	

<table>
<tr><td rowspan="4">The upper limit size of the memory usage [Mbytes]</td><td colspan="3">Specify the memory size to be used for reading the debug information[Note].</td></tr>
<tr><td>Default</td><td colspan="2">500</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Decimal number between 100 and 1000</td></tr>
<tr><td rowspan="5">Search for global variables even outside the current scope</td><td colspan="3">Specify whether searching for global variables also proceeds outside the scope of the current PC value when those variables for which the scope has not been specified are registered in a watch-expression.
Note: Specifying [Yes] for this property may lower the response speed of the debug tool.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Searching for global variables proceeds outside the current scope.</td></tr>
<tr><td>No</td><td>Searching for global variables only proceeds within the current scope.</td></tr>
</table>

Note In some cases, lowering the upper limit may lead to poorer responsiveness since it leads to more frequent discarding and re-reading of debug information.

[Flash Options Settings] tab

This tab is used to configure options for the flash memory incorporated in the microcontroller.
Note that this tab appears only when the selected microcontroller supports the flash options.

Caution 1. [Full-spec emulator][E1][E20]
You can configure options only while connected to the debug tool.
[Simulator]
You can configure options only while disconnected from the debug tool.

Caution 2. CPU reset may be generated automatically depending on the selected microcontroller when you change the configuration on this tab.

(1) [Flash Options]

[Description of each category]

(1) [Flash Options]
The detailed information on the flash options is displayed and its configuration can be changed.

Flash options	Specify options for the flash memory.	
	Default	Flash options
	Modifying	Specify with the Flash Options Setting dialog box. The Flash Options Setting dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify options for the flash memory on this panel). Note that the contents of options for the flash memory that have been specified are not displayed on this panel.

[Hook Transaction Settings] tab

This tab is used to display the detailed information categorized by the following and the configuration can be changed.
For details on the hook transaction, see "2.20 Use Hook Function".

(1) [Hook Transaction Settings]

[Description of each category]

(1) [Hook Transaction Settings]
The detailed information on the hook transaction is displayed and its configuration can be changed.

Before download	Specify the process[Note] to proceed right before downloading the load module file.	
	Default	Before download[0] ("[]" is the current number of specified processes.)
	Modifying	Specify with the Text Edit dialog box. The Text Edit dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify options for the flash memory on this panel).
	Available values	Up to 128 processes (one line in the Text Edit dialog box is equivalent to one processing) Note that up to 64 characters for one process can be specified.
After download	Specify the process[Note] to proceed right after downloading the load module file.	
	Default	Before download[0] ("[]" is the current number of specified processes.)
	Modifying	Specify with the Text Edit dialog box. The Text Edit dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify options for the flash memory on this panel).
	Available values	Up to 128 processes (one line in the Text Edit dialog box is equivalent to one processing) Note that up to 64 characters for one process can be specified.
After CPU reset under breaking	Specify the process[Note] to proceed right after CPU reset during break.	
	Default	Before download[0] ("[]" is the current number of specified processes.)
	Modifying	Specify with the Text Edit dialog box. The Text Edit dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify options for the flash memory on this panel).
	Available values	Up to 128 processes (one line in the Text Edit dialog box is equivalent to one processing) Note that up to 64 characters for one process can be specified.
Before running	Specify the process[Note] to proceed right before the execution of the program.	
	Default	Before download[0] ("[]" is the current number of specified processes.)
	Modifying	Specify with the Text Edit dialog box. The Text Edit dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify options for the flash memory on this panel).
	Available values	Up to 128 processes (one line in the Text Edit dialog box is equivalent to one processing) Note that up to 64 characters for one process can be specified.

After breaking	Specify the process[Note] to proceed right after the program break.	
	Default	Before download[0] ("[]" is the current number of specified processes.)
	Modifying	Specify with the Text Edit dialog box. The Text Edit dialog box is opened by clicking the [...] button that appears at right edge of this field when you select this property (you cannot specify options for the flash memory on this panel).
	Available values	Up to 128 processes (one line in the Text Edit dialog box is equivalent to one processing) Note that up to 64 characters for one process can be specified.

Note From the following three processes, input the specification format of the desired process to the Text Edit dialog box.

[Process 1]:

Automatically overwrites the value of *I/O register* with *Value*.

Specification format:

```
I/O-register-name Value
```

[Process 2]:

Automatically overwrites the value of *CPU register* with *Value*.

Specification format:

```
CPU-register-name Value
```

[Process 3]:

Automatically executes a script file which is specified with *Python script path* (absolute path or relative path from the project folder).

Specification format:

```
Source Python-script-path
```

Memory panel

This panel is used to display the contents of the memory and change the memory value (see "2.12.1 Display/change the memory").

Furthermore, the contents of data flash memory (including ID tag) can be displayed and changed when the selected microcontroller incorporates the data flash memory.

Up to a maximum of four of these panels can be opened. Each panel is identified by the names "Memory1", "Memory2", "Memory3", and "Memory4" on the titlebar.

The display contents are automatically updated when the value of the memory changes after a program is executed (when the execution is done in steps, the display is updated after each step). In addition, by enabling the Real-time display update function, it is also possible to update the display contents in real-time even while a program is being executed.

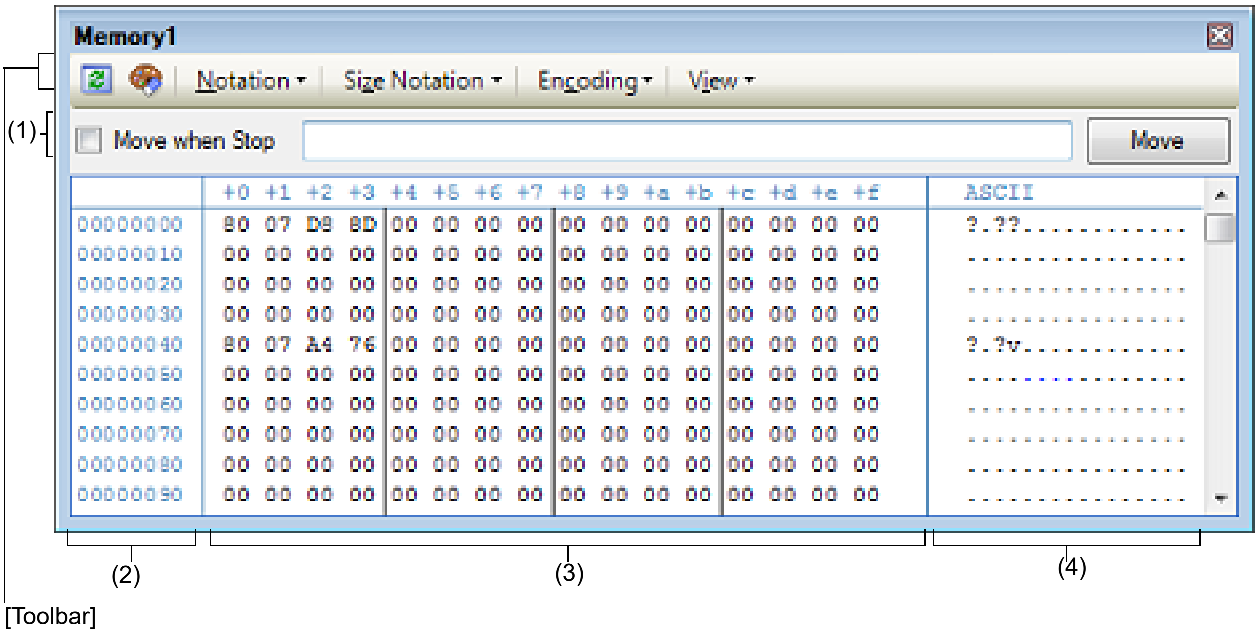
This panel appears only when connected to the debug tool.

Caution CPU reset may be generated depending on the selected microcontroller if you change the memory value in the data flash area.

Remark 1. This panel can be zoomed in and out by 100% in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Remark 2. You can set the scroll range of the vertical scroll bar on this panel via the Scroll Range Settings dialog box which is opened by clicking the button from [View] on the toolbar.

Figure A.6 Memory Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Memory panel-dedicated items)]
- [[Edit] menu (Memory panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Memory] >> [Memory*1-4*].

[Description of each area]

(1) Display position specification area
It is possible to specify the display start position of the memory contents by specifying an address expression.
Specify the following items.

(a) Specify an address expression
Directly input the address expression of the memory value address to display in the text box. You can specify an input expression with up to 1024 characters. The result of the expression is treated as the display start position address.
Note that an address value greater than the microcontroller address space cannot be specified.
In addition, an address value greater than the value expressed within 32 bits cannot be specified.

Remark 1. A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (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.

(b) Specify automatic/manual evaluation of the address expression
The timing to change the display start position can be determined by specifying in the [Move when Stop] check box and the [Move] button.

[Move when Stop]		The caret is moved to the address which is automatically calculated from the address expression after the program is stopped.
		The address expression is not automatically evaluated after the program is stopped. Click the [Move] button to manually evaluate the address expression.
[Move] button		When the [Move when Stop] check box is not checked, click this button to evaluate the address expression and move the caret to the result address of the evaluation.

(2) Address area
The address of the memory is displayed (hexadecimal number notation fixing).
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 corresponds to the one in memory space of the specified microcontroller in the project.
This area cannot be edited.

Caution The offset value that have been set is automatically changed in accordance with the number of view columns in the Memory value area.

(3) Memory value area
The value of the memory is displayed and changed.
Specification of the display notation, display width of memory values or the number of view columns is performed by selecting the buttons on the toolbar or [Notation]/[Size Notation]/[View] from the context menu (see "2.12.1.2 Change display format of values").
The meanings of the marks and colors displayed as memory values are as follows (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box):

Display Example (Default)			Description
00	Character color	Blue	Memory value that the user is changing Press the [Enter] key to write to the target memory.
	Background color	Standard color	
00 (Under line)	Character color	Standard color	Memory value of the address whose symbol has been defined. Watch-expressions can be registered to this memory (see "(f) Registering watch-expressions").
	Background color	Standard color	
00	Character color	Brown	Memory value that has been changed because of the execution of a program[Note] To reset the highlighting, select the  button on the toolbar.
	Background color	Cream	

<table>
<tr><th colspan="3">Display Example (Default)</th><th colspan="2">Description</th></tr>
<tr><td rowspan="2">00</td><td>Character color</td><td>Pink</td><td colspan="2" rowspan="2">Memory value for which the Real-time display update function is being operated</td></tr>
<tr><td>Background color</td><td>Standard color</td></tr>
<tr><td rowspan="2">00</td><td>Character color</td><td>Standard color</td><td rowspan="2">Read/Fetch</td><td rowspan="6">Current access condition of the memory value when the Real-time display update function is being operated</td></tr>
<tr><td>Background color</td><td>Palegreen</td></tr>
<tr><td rowspan="2">00</td><td>Character color</td><td>Standard color</td><td rowspan="2">Write</td></tr>
<tr><td>Background color</td><td>Orange</td></tr>
<tr><td rowspan="2">00</td><td>Character color</td><td>Standard color</td><td rowspan="2">Read and Write</td></tr>
<tr><td>Background color</td><td>Paleturquoise</td></tr>
<tr><td rowspan="2">00</td><td>Character color</td><td>Gray</td><td colspan="2" rowspan="2">Memory value of the read-protected area</td></tr>
<tr><td>Background color</td><td>Standard color</td></tr>
<tr><td rowspan="2">??</td><td>Character color</td><td>Gray</td><td colspan="2" rowspan="2">Areas not memory-mapped</td></tr>
<tr><td>Background color</td><td>Standard color</td></tr>
<tr><td rowspan="2">--</td><td>Character color</td><td>Gray</td><td colspan="2" rowspan="2">Areas not rewritable (e.g. I/O register area/I/O protection area) or when acquisition of memory values failed</td></tr>
<tr><td>Background color</td><td>Standard color</td></tr>
<tr><td rowspan="2">**</td><td>Character color</td><td>Standard color</td><td colspan="2" rowspan="2">When display is specified for other than the real-time display update area during program execution or when acquisition of memory values failed</td></tr>
<tr><td>Background color</td><td>Standard color</td></tr>
</table>

Note Just before execution of a program, only the memory value in the address range for which the Memory panel had been displayed becomes the target.
In addition, the value is not highlighted if it is same for before and after the execution of the program.

Caution The number of view columns is automatically changed in accordance with the set value of [Size Notation] of the context menu.

This area is provided with the following functions.

(a) Pop-up display
The following contents are pop-up displayed based on the nearest existing symbol forward from the address the mouse is designating when hovering the mouse cursor over the memory value.
Note that if there is no symbol information (the underlining is non-display), no pop-up display is done.

variable + 0x14

Symbol name Offset value

Symbol name	Indicates the name of the symbol.
Offset value	When a symbol has not been defined for the addresses, the offset value from the nearest symbol exists forward is displayed (hexadecimal number notation fixing).

(b) Real-time display update function
Using the real-time display update function allows you to display/modify the value of the memory contents not only while the program is stopped, but also in execution.
See "2.12.1.4 Display/modify the memory contents during program execution" for details on the real-time display update function.

(c) Changing memory values
Directly edit from the keyboard after moving the caret to the memory value to be edited.
The color of the memory value changes when it is in editing. Press the [Enter] key to write the edited value to the target memory (if the [Esc] key is pressed before the [Enter] key is pressed, the editing is cancelled).
See "2.12.1.3 Modify the memory contents" for details on the method for changing the memory value.

(d) Searching/initializing memory value
The Memory Search dialog box is opened to search the memory contents in the specified address range by selecting [Find...] from the context menu (see "2.12.1.5 Search the memory contents").
In addition, the Memory Initialize dialog box is opened to change the memory contents collectively in the specified address range by selecting [Fill...] from the context menu (see "2.12.1.6 Modify the memory contents in batch (initialize)").

(e) Copying and pasting
By selecting a range of memory values with the mouse, the contents of the range can be copied to the clipboard as a character string, and these contents can be pasted to the caret position.
These operations are performed by selecting from the context menu or selecting from the [Edit] menu.
However, the paste operation is possible only when the character string to be pasted and the display notation (radix and size) of the area match.
If the display notation does not match, a message is displayed.
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) Registering watch-expressions
A memory value with underline indicates that a symbol has been defined in the address, and its symbol can be registered as a watch-expression.
After selecting the memory value or placing the caret on the memory value, the symbol name of the address is registered in the Watch panel (Watch1) as a watch-expression by selecting [Register to Watch1] from the context menu.

Caution A memory value without underline cannot be registered as a watch-expression.

(g) Saving the contents of memory values
The Data Save dialog box can be opened by selecting the [File] menu >> [Save Memory Data As...], and the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.12.1.7 Save the memory contents" for details on the method for saving the contents of memory values.

(4) Character strings area
Memory values converted into character code are displayed.
The character code can be specified by selecting [Encoding] from the toolbar or 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	

Item	Display Format		Size
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 bytes of the number of view columns".

This area is provided with the following functions.

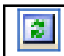
(a) Changing character strings
Directly edit from the keyboard after moving the caret to the character string to be edited.
The color of the character string changes when it is in editing. Press the [Enter] key to write the edited value to the target memory (if the [Esc] key is pressed before the [Enter] key is pressed, the editing is cancelled).

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

(b) Searching character strings
The Memory Search dialog box is opened to search for character strings by selecting [Find...] from the context menu (see "2.12.1.5 Search the memory contents").

(c) Copying and pasting
By selecting a range of character strings with the mouse, the contents of the range can be copied to the clipboard as a character string, and these contents can be pasted to the caret position.
These operations are performed by the selecting from the context menu or selecting from the [Edit] menu.
However, the paste operation is possible only when [ASCII] has been selected as the character code. If other than [ASCII] is selected, a message is displayed.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel.
	Resets highlighting of values that have been changed by executing a program. This item is disabled during execution of a program.
Notation	The following buttons to change the notation of memory values are displayed.

Hexadecimal	Displays memory values in hexadecimal number (default).
Signed Decimal	Displays memory values in signed decimal number.
Unsigned Decimal	Displays memory values in unsigned decimal number.
Octal	Displays memory values in octal number.
Binary	Displays memory values in binary number.
Size Notation	The following buttons to change the notation of sizes 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.
Encoding	The following buttons to change the encoding of 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 Big-Endian	Displays character strings in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character strings in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character strings in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character strings in UTF-32 Little-Endian 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	The following buttons to change the display format are displayed.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range for this panel.
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.

[[File] menu (Memory panel-dedicated items)]

The following items are exclusive for the [File] menu in the Memory panel (other items are common to all the panels). Note that all these items are disabled during execution of a program.

Save Memory Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(g) Saving the contents of memory values"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Memory Data As...].
Save Memory Data As...	Opens the Data Save dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(g) Saving the contents of memory values").

[[Edit] menu (Memory panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Memory panel (all other items are disabled). Note that all these items are disabled during execution of a program.

Copy	Copies the contents of the selected range to the clipboard as character string(s).
Paste	Pastes the character string(s) copied in the clipboard to the caret position. To the memory value area: See "(e) Copying and pasting". To the character strings area: See "(c) Copying and pasting".
Find...	Opens the Memory Search dialog box. The search is operated either in the Memory value area or the Character strings area, in which a caret is.

[Context menu]

Register to Watch1	Registers the symbol at the caret to the Watch panel (Watch1). At this time, since it is registered as a variable name, the symbol name that is displayed changes depending on the scope. Note that this item is disabled when no symbol has been defined in the address corresponding to the memory value at the caret position (see "(f) Registering watch-expressions").
Find...	Opens the Memory Search dialog box. The search is operated either in the Memory value area or the Character strings area (unless the floating-point value display is selected), in which a caret is. This item is disabled during execution of a program.
Fill...	Opens the Memory Initialize dialog box.
Refresh	Acquires the latest data from the debug tool, and updates the contents of this panel.
Copy	Copies the contents of the selected range to the clipboard as character string(s). This item is disabled during execution of a program.
Paste	Pasts the character string(s) copied in the clipboard to the caret position. This item is disabled during execution of a program. To the memory value area: See "(e) Copying and pasting". To the character strings area: See "(c) Copying and pasting".

Notation	The following cascade menus are displayed to specify the notation of memory values.
Hexadecimal	Displays memory values in hexadecimal number (default).
Signed Decimal	Displays memory values in signed decimal number.
Unsigned Decimal	Displays memory values in unsigned decimal number.
Octal	Displays memory values in octal number.
Binary	Displays memory values in binary number.
Size Notation	The following cascade menus are displayed to specify the notation of sizes of memory values.
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.
Encoding	The following cascade menus are displayed to specify the display format in the character strings area.
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 Big-Endian	Displays character strings in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character strings in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character strings in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character strings in UTF-32 Little-Endian 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	The following cascade menus are displayed to specify the display format.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range for this panel.
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	Highlights memory values that have changed by execution of a program if this item is checked (default). This item is disabled during execution of a program.
Periodic Updating	The following cascade menus are displayed to set for the real-time display update function (see "(b) Real-time display update function").
Periodic Updating Options	Opens the Property panel to set for the real-time display update function.

Disassemble panel

This panel is used to display the results of disassembling the contents of the memory (disassembled text), and execute line assembly (see "2.7.4 Perform line assembly").

Furthermore, the instruction level debugging (see "2.10.3 Execute programs in steps") and the code coverage measurement result display **[Simulator]** (see "2.17 Measure Coverage [Simulator]") can be performed in this panel.

Up to a maximum of four of these panels can be opened. Each panel is identified by the names "Disassemble1", "Disassemble2", "Disassemble3" and "Disassemble4" on the titlebar.

The source text in the source file corresponding to the code data can also be displayed by setting to the Mixed display mode (default).

This panel appears only when connected to the debug tool.

Caution A step execution is performed in instruction level units when the focus is in this panel (see "2.10.3 Execute programs in steps").

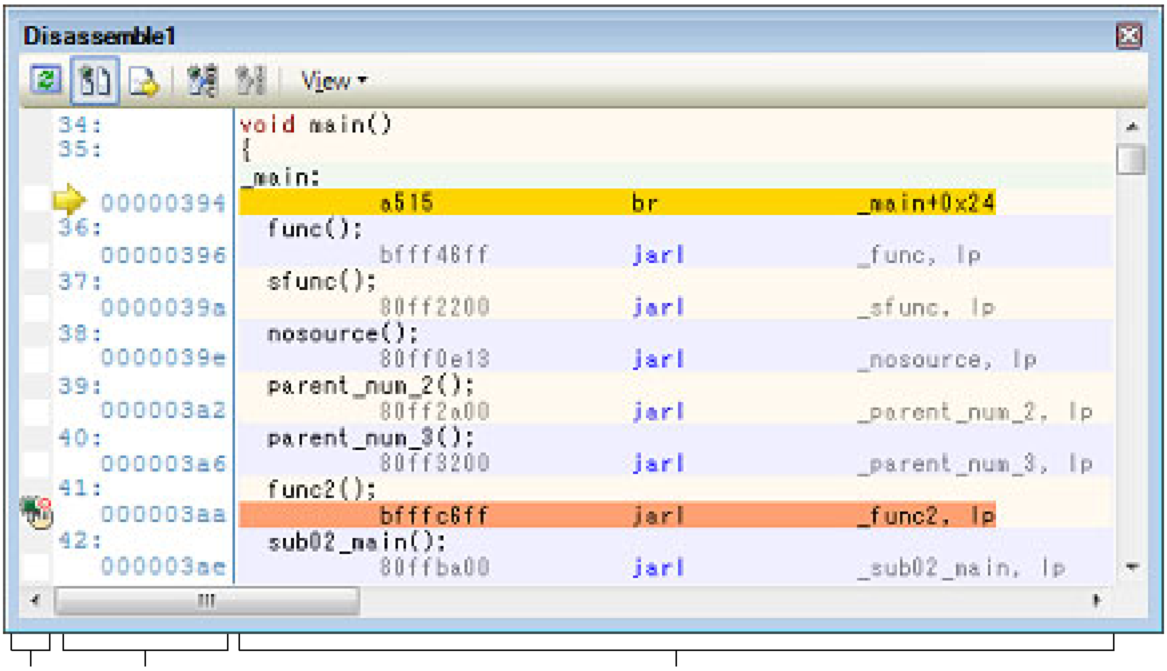
Remark 1. You can set the scroll range of the vertical scroll bar on this panel via the Scroll Range Settings dialog box which is opened by clicking the button from [View] on the toolbar.

Remark 2. You can print the current screen image of this panel by selecting [Print...] from the [File] menu.

Remark 3. This panel can be zoomed in and out by 100% in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Figure A.7 Disassemble Panel (When Mixed Display Mode Is Selected)

[Tool bar]



(1) (2) (3)

Figure A.8 Disassemble Panel (When Disassemble Display Mode Is Selected)

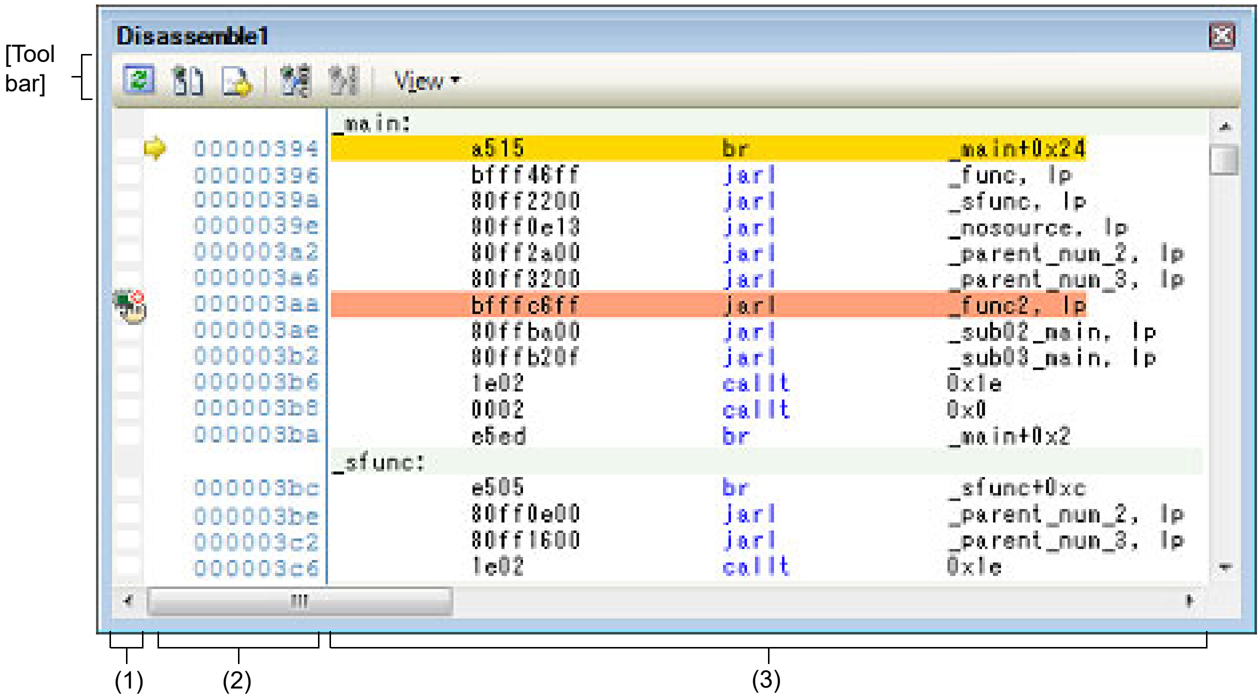
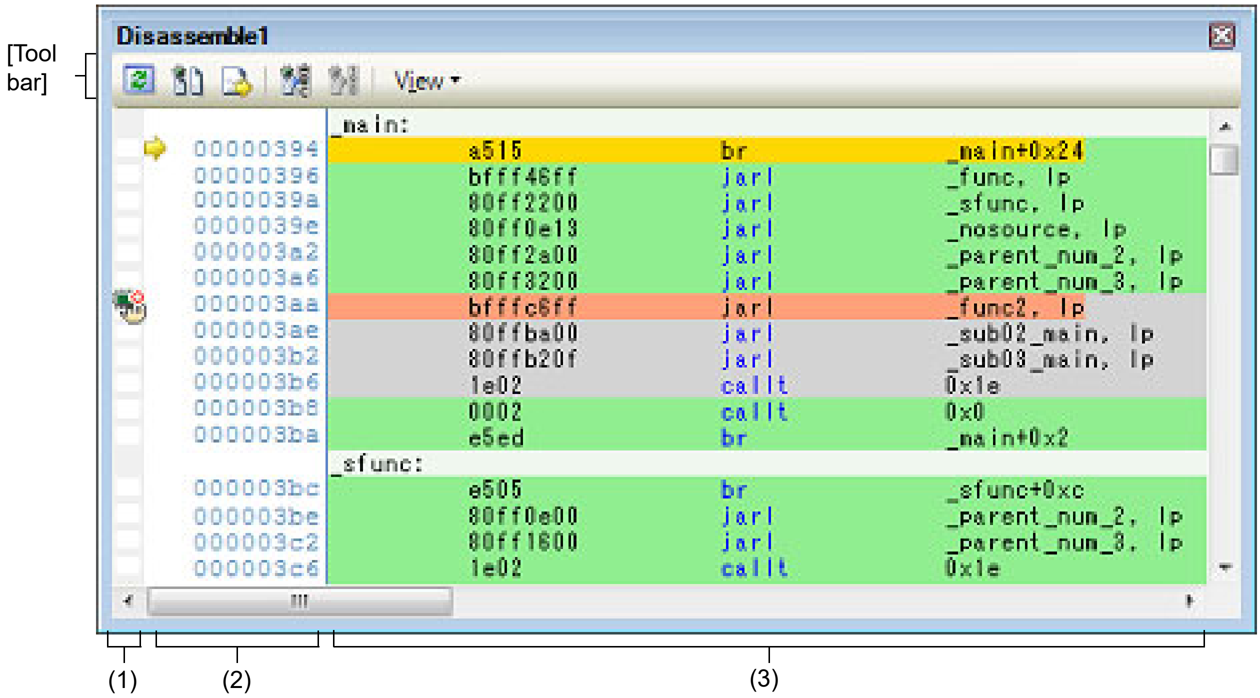


Figure A.9 Disassemble Panel (When Code Coverage Measurement Result Is Displayed) [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Disassemble panel-dedicated items)]
- [[Edit] menu (Disassemble panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Disassemble] >> [Disassemble*1 - 4*].

[Description of each area]

(1) Event area

The lines for which events can be set are shown with the background color in white (this mean that events cannot be set for those lines whose background color in gray).
In addition, the Event mark corresponding to an event that has been currently set is displayed.
This area is provided with the following functions.

(a) Setting/deleting breakpoints

By clicking where you want to set a breakpoint with the mouse, the breakpoint can be set easily.
The breakpoint is set to the instruction at the start address of the clicked line.
Once the breakpoint is set, the Event mark is displayed at the line that is set. In addition, the detailed information about the set breakpoint is reflected in the Events panel.
When this operation is performed at a place where any one of the event marks is already being displayed, that event is deleted and the setting of breakpoints cannot be done.
Note that the setting of events can be done only for those lines where the background color is shown in white.
See "2.11.3 Stop the program at the arbitrary position (breakpoint)" for details on how to set the breakpoint.

(b) Changes event status

Event status can be changed from the following menu displayed by right-clicking the event mark.

Enable Event	Changes the selected event state to a Valid state. Event occurs when the specified condition is met. When the event mark ( ) which indicates that multiple events have been set is selected, all of the events that have been set are enabled.
Disable Event	Changes the selected event state to an Invalid state. Event does not occur when the specified condition is met. When the event mark ( ) which indicates that multiple events have been set is selected, all of the events that have been set are disabled.
Delete Event	Deletes the selected event. When the event mark ( ) which indicates that multiple events have been set is selected, all of the events that have been set are deleted.
View Event Detailed Setup	Opens the Events panel to display the detailed information of the selected event.

(c) Pop-up display

By hovering the mouse cursor over the Event mark, the name of the event, the detailed information for the event and the comments added to the event are pop-up displayed.
When multiple events have been set in the applicable place, information for each event, up to a maximum of three events, is listed and displayed.

(2) Address area

The address per line to start disassembling is displayed (hexadecimal number notation fixing).
In addition, the current PC mark ( ) that corresponds to the current PC position (PC register value) is displayed.
The address width corresponds to the one in memory space of the specified microcontroller in the project.
For the source text line in the Mixed display mode, line numbers (*xxx*:) in the source file correspond to the start address are displayed.

Remark The current PC mark changes from to when the position of the current PC is invalid (e.g., in cases where the current target core for debugging differs from the one selected at the time the program was stopped).

This area is provided with the following functions.

(a) Pop-up display

By hovering the mouse cursor over a address or line number, the following information is pop-up displayed.

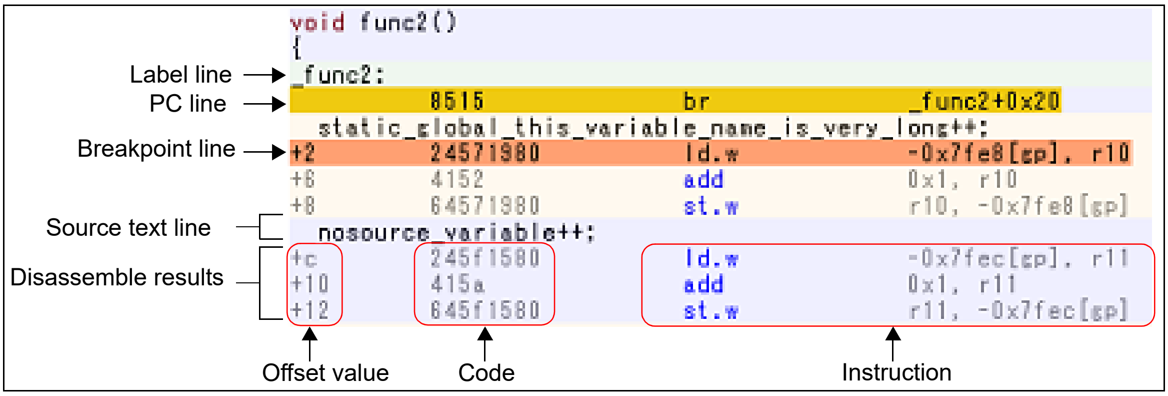
Address	Format:<*Label name*> + <*Offset value*> Example1: main + 0x10 Example2: sub function + 0x20
Source line number	Format:<*Load module name*>[Note]$<*File name*> # <*Line number*> Example1: test1.out$main.c#40 Example2: main.c#100

Note <*Load module name*> is displayed only when multiple load modules have been downloaded to the debug tool.

(3) Disassemble area

The results of disassembling are displayed next to the corresponding source text as follows:

Figure A.10 Display Contents of Disassemble Area (In Case of Mixed Display Mode)



Label line		The label is displayed when a label is defined for the address, and its corresponding line is shown highlighted in lightgreen.
PC line		A line corresponding to an address of the current PC (PC register value) is shown highlighted[Note 1].
Breakpoint line		A line at which a breakpoint is set is shown highlighted[Note 1].
Source text line		The source text corresponding to the code data is displayed[Note 2].
Disassemble results	Offset value	The offset value from the nearest label is displayed when a label is defined for the address[Note 3].
	Code	The code that is the target of disassembly is displayed in hexadecimal number.
	Instruction	Instruction is displayed as the result of disassembling. The mnemonics are shown highlighted in blue.

Note 1. The highlighting color depends on the configuration in the [General - Font and Color] category of the Option dialog box.

Note 2. The source text can be set to non-display by clicking the button (toggle) on the toolbar or removing the check for [Mixed Display] from the context menu (this option is checked by default).

Note 3. Offset values are not displayed by default. They can be displayed by clicking the button on the toolbar or selecting [Show Offset] from the context menu.

This area is provided with the following functions.

(a) Line assembly

Instructions and code displayed in this panel can be edited (line assembly).

See "2.7.4 Perform line assembly" for details on how to operate it.

(b) Program execution by instruction level

Execution can be controlled at the instruction level unit by step executing a program in a state where there is a focus on this panel.

See "2.10.3 Execute programs in steps" for details on how to operate it.

(c) Setting of various events
Various events can be set to the addresses/lines where the caret currently exists by selecting [Break Settings], [Trace Settings], [Timer Settings] or [Performance Measurement Settings] from the context menu.
The corresponding Event mark is displayed in the Event area when an event is set. In addition, the detailed information about the set event is reflected in the Events panel.
Note, however, that the setting of events can be done only for those lines where the background color is shown in white in the event area.
See the following for details on how to set events.

- "2.11.4 Stop the program at the arbitrary position (break event)"
- "2.11.5 Stop the program with the access to variables/I/O registers"
- "2.14.3 Collect execution history in a section"
- "2.14.4 Collect execution history only when the condition is met"
- "2.15.2 Measure execution time in a section"
- "2.16.1 Measure the performance in a section"
- "2.18.1 Inset printf"

Remark A breakpoint can be set or deleted easily in the Event area as well (see "(a) Setting/deleting breakpoints").

(d) Registering a watch-expression
Variable names of C language, CPU registers, I/O registers, and assembler symbols can be registered in the Watch panel as watch-expressions.
See "2.12.6.1 Register a watch-expression" for details on how to operate it.

(e) Moving to symbol definition place
By clicking the button on the toolbar or selecting [Go to Symbol] from the context menu in a state where the caret has been moved to a instruction that has referenced a symbol, the caret position is moved to the address where the symbol at the caret position has been defined.
In addition, when following on this operation you click on the button on the toolbar or select [Back to *Address*] from the context menu, the caret position is returned to the instruction that has referenced a symbol before the caret was moved (the address value of the instruction that has referenced a symbol is displayed in *Address*).

(f) Jump to source line and memory
Selecting [Jump to Source] from the context menu will open the Editor panel with the caret moved to the source line corresponding to the address at the current caret position (if the Editor panel is already open, the screen will jump to the panel).
In addition, similarly, selecting [Jump to Memory] will open the Memory panel (Memory1) with the caret moved to the memory value corresponding to the address at the current caret position (if the Memory panel (Memory1) is already open, the screen will jump to the panel).

(g) Code coverage measurement result display [Simulator]
When the coverage function is valid, lines corresponding to the specified coverage measurement area are shown highlighted based on the code coverage measurement result that is acquired by executing the program.
See "2.17 Measure Coverage [Simulator]" for details on the coverage measurement.

(h) Saving the contents of disassembled data
The Data Save dialog box can be opened by selecting the [File] menu >> [Save Disassemble Data As…], and the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.7.2.5 Save the disassembled text contents" for details on the method for saving the contents of disassembled data.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel.
	Toggles between the mixed display mode (default) and disassemble display mode, as the display mode of this panel (see "2.7.2.1 Change display mode").
	Specifies the caret position so that it follows the current PC value.

	Moves the caret to the define position of the selected symbol.
	Moves the caret to the position (*address*) immediately before it is moved with the button.
View	The following buttons to set the display contents in the disassemble area are displayed.
Show Offset	Displays the offset value of the label. The offset value from the nearest label is displayed when a label is defined for the address.
Show Symbol	Displays the address value in the format "symbol + offset value" (default). Note that when a symbol has been defined as the address value, only the symbol is displayed.
Show Function Name	Displays the name of the register by its function name (default).
Show Absolute Name	Displays the name of the register by its absolute name.
	Opens the Scroll Range Settings dialog box to set the scroll range for this panel.

[[File] menu (Disassemble panel-dedicated items)]

The following items are exclusive for the [File] menu in the Disassemble panel (other items are common to all the panels).

Note that all these items are disabled during execution of a program.

Save Disassemble Data	Overwrites the contents of the disassembling to the previously saved text file (*.txt)/CSV file (*.csv) (see "(h) Saving the contents of disassembled data"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Disassemble Data As…].
Save Disassemble Data As…	Opens the Data Save dialog box to newly save the contents of the disassembling to the specified text file (*.text)/CSV file (*.csv) (see "(h) Saving the contents of disassembled data").
Print...	Opens the Print Address Range Settings dialog box for printing the contents of this panel.

[[Edit] menu (Disassemble panel-dedicated items)]

The following items are exclusive for the [Edit] menu in the Disassemble panel (all other items are disabled).

Copy	When a line is selected, copies the contents of the selected line to the clipboard as a character string. In the case of the edit mode, copies the selected character string to the clipboard.
Rename	Changes to the edit mode to edit the instruction/code at the caret position (see "2.7.4 Perform line assembly"). This item is disabled during execution of a program.
Find…	Opens the Find and Replace dialog box with selecting the [Find in Files] tab.
Replace…	Opens the Find and Replace dialog box with selecting the [Replace in Files] tab.
Move…	Opens the Go to the Location dialog box to move the caret to the specified address.

[Context menu]

(1) Disassemble area and Address area

Register to Watch1	Registers the selected character string or the word at the caret position to the Watch panel (Watch1) as a watch-expression (the judgment of the word depends on current build tool). At this time, since it is registered as a variable name, the symbol name that is displayed changes depending on the scope.
Register Action Event...	Opens the Action Events dialog box to set an action event to the address at the caret position.
Go to Here	Executes the program from the address indicated by the current PC value to the address corresponding to the line at the caret position. This item is disabled during program execution/build (not including rapid build) execution.
Set PC to Here	Sets the address of the line at the current caret position to the current PC value. This item is disabled during program execution/build (not including rapid build) execution.
Move…	Opens the Go to the Location dialog box to move the caret to the specified address.
Go to Symbol	Moves the caret to the define position of the selected symbol.
Back to *Address*	Moves the caret to the position (*address*) immediately before it is moved by [Go to Symbol]. Note that this item is disabled when no symbol name is displayed in the address.
Break Settings	The following cascade menus are displayed to set the break-related event.
Set Hardware Break	Sets a breakpoint (Hardware Break event to the address at the caret position (see "2.11.3 Stop the program at the arbitrary position (breakpoint)").
Set Software Break [Full-spec emulator][E1][E20]	Sets a breakpoint (Software Break event) to the address at the caret position (see "2.11.3 Stop the program at the arbitrary position (breakpoint)").
Set Combination Break	In this product version, this item is not supported.
Set Read Break to	Sets a break event with read access condition to a variable at the caret or a selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "2.11.4 Stop the program at the arbitrary position (break event)").
Set Write Break to	Sets a break event with write access condition to a variable at the caret or a selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "2.11.4 Stop the program at the arbitrary position (break event)").
Set R/W Break to	Sets a break event with read/write access condition to a variable at the caret or a selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register (see "2.11.4 Stop the program at the arbitrary position (break event)").
Break Option	Opens the Property panel to set the break function.

Trace Settings	The following cascade menus are displayed to set the trace-related event.
Start Tracing	Sets a trace start event to start collecting the trace data when an instruction of an address at the caret position is executed (see "2.14.3 Collect execution history in a section")Note 1.
Stop Tracing	Sets a trace end event to stop collecting the trace data when an instruction of an address at the caret position is executed (see "2.14.3 Collect execution history in a section")Note 1.
Record Reading Value	Sets a Point Trace event to record the access value as the trace data when a variable at the caret or the selected variable (global variable, static variable inside functions, file-internal static variable) or I/O register is read accessed (see "2.14.4 Collect execution history only when the condition is met").
Record Writing Value	Sets a Point Trace event to record the access value as the trace data when a variable at the caret or the selected variable (global variable, static variable inside functions, file-internal static variable) or I/O register is write accessed (see "2.14.4 Collect execution history only when the condition is met").
Record R/W Value	Sets a Point Trace event to record the access value as the trace data when a variable at the caret or a selected variable (global variable/static variable inside functions/file-internal static variable)/I/O register is read/write accessed (see "2.14.4 Collect execution history only when the condition is met").
Record Start R/W Value [E1][E20]	Sets a trace start event to start collecting the trace data when a variable at the caret or the selected variable (global variable, static variable inside functions, file-internal static variable) /I/O register is read/ write accessed (see "2.14.3.1 Set a Trace event").
Record End R/W Value [E1][E20]	Sets a trace end event to stop collecting the trace data when a variable at the caret or the selected variable (global variable, static variable inside functions, file-internal static variable) /I/O register is read/ write accessed (see "2.14.3.1 Set a Trace event").
Show Trace Result	Opens the Trace panel and displays the acquired trace data.
Trace Settings	Opens the Property panel to set the trace function.
Timer Settings	The following cascade menus are displayed to set the timer-related event (see "2.15.2 Measure execution time in a section").
Start timer	Sets a timer start event to start measuring the execution time of the program when an instruction of an address at the caret position is executedNote 2.
Set Timer *n*	Specify a channel (*n*: 1 to 8) in which a timer start event is set.
Stop timer	Sets a timer end event to stop measuring the execution time of the program when an instruction of an address at the caret position is executedNote 2.
Set Timer *n*	Specify a channel (*n*: 1 to 8) in which a timer stop event is set.
View Result of Timer	Opens the Events panel and displays only timer-related events.

Performance Measurement Settings [Full-spec emulator] [E1/E20]	The following cascade menus are displayed to set the performance measurement-related event.
Start Performance Measurement	Sets a performance measurement start event to start measuring performance measurement when an instruction of an address at the caret position is executed.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Stop Performance Measurement	Sets a performance measurement end event to stop measuring performance measurement when an instruction of an address at the caret position is executed.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
Set Performance Measurement Start Read Value	Sets a performance measurement start event that causes performance measurement to start upon 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.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Set Performance Measurement End Read Value	Sets a performance measurement end event that causes performance measurement to stop upon 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.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
Set Performance Measurement Start Write Value	Sets a performance measurement start event that causes performance measurement to start upon 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.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Set Performance Measurement End Write Value	Sets a performance measurement end event that causes performance measurement to stop upon 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.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
Set Performance Measurement Start R/W Value	Sets a performance measurement start event that causes performance measurement 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.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Set Performance Measurement End R/W Value	Sets a performance measurement end event that causes performance measurement to stop 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.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
View Result of Performance Measurement	Opens the Events panel and displays only performance measurement-related events.
Clear Coverage Information [Simulator]	Clears all the coverage measurement results currently being stored in the debug tool.

Edit Disassemble	Changes to the edit mode to edit the instruction of the line at the caret position (see "2.7.4 Perform line assembly"). This item is disabled during execution of a program.
Edit Code	Changes to the edit mode to edit the code of the line at the caret position (see "2.7.4 Perform line assembly"). This item is disabled during execution of a program.
View	The following cascade menus to set the display contents in the disassemble area are displayed.
Show Offset	Displays the offset value of the label. The offset value from the nearest label is displayed when a label is defined for the address.
Show Symbol	Displays the address value in the format "symbol + offset value" (default). Note that when a symbol has been defined as the address value, only the symbol is displayed.
Show Function Name	Displays the name of the register by its function name (default).
Show Absolute Name	Displays the name of the register by its absolute name.
Settings Scroll Range...	Opens the Scroll Range Settings dialog box to set the scroll range for this panel.
Mixed Display	Toggles between the mixed display mode (default) and disassemble display mode, as the display mode of this panel (see "2.7.2.1 Change display mode").
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the address at the caret position in this panel.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the memory value corresponding to the address at the caret position in this panel.

Note 1. **[Simulator]**
The [Use trace function] property in the [Trace] category on the Property panel is automatically set to [Yes].

Note 2. **[Simulator]**
The [Use timer function] property in the category on the Property panel is automatically set to [Yes].

(2) Event area **[Full-spec emulator][E1][E20]**

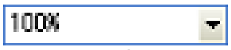
Hardware Break First	The type of break that can be set by a one click operation of the mouse is set as a hardware breakpoint (this is reflected in the setting of the [First using type of breakpoint] property in the [Break] [Full-spec emulator][E1][E20] category on the Property panel).
Software Break First	The type of break that can be set by a one click operation of the mouse is set as a software breakpoint (this is reflected in the setting of the [First using type of breakpoint] property in the [Break] [Full-spec emulator][E1][E20] category on the Property panel).

CPU Register panel

This panel is used to display the contents of the CPU register (program registers/system registers) and change the CPU register values (see "2.12.2 Display/change the CPU register").

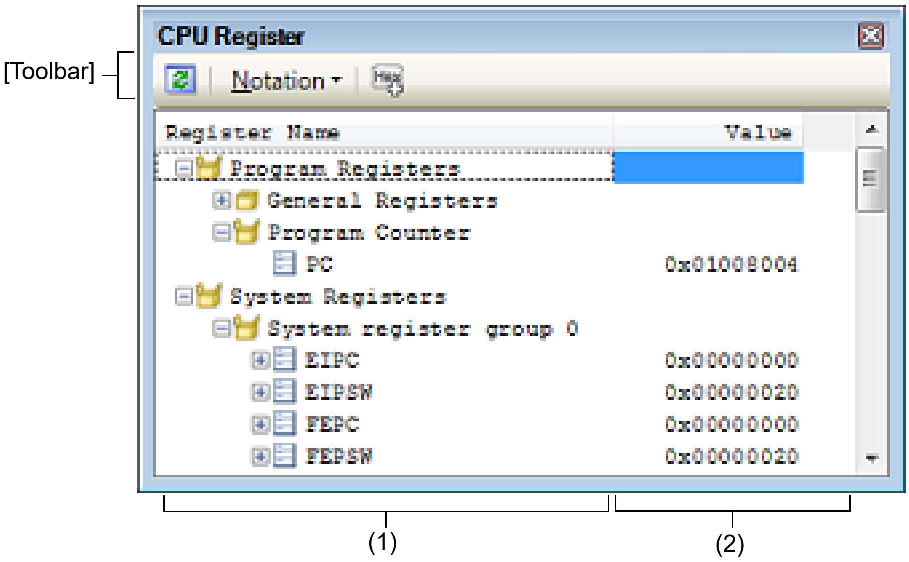
This panel appears only when connected to the debug tool.

Caution When the selected microcontroller supports multi-core, this panel displays/changes the value regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

Remark 1. This panel can be zoomed in and out by  in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Note 1. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A.11 CPU Register Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (CPU Register panel-dedicated items)]
- [[Edit] menu (CPU Register panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [CPU Register].

[Description of each area]

(1) [Register Name] area

The types of register are classified as categories (folders), and a list of the respective register names is displayed. Note that neither category names nor register names can be edited and deleted.

The meanings of the icons are as follows:

	Indicates that the register name belonging to this category is displayed. When you double-click on the icon, or click on the "-" mark, the category is closed and the register name is hidden.

	Indicates that the register name belonging to this category is hidden. When you double-click on the icon, or click on the "+" mark, the category is opened and the register name is displayed.
	Indicates the name of the register. When you double-click on the icon, or click on the "+" or "-" marks, the name of the register part is displayed or hidden.
	Indicates the name of the register part.

This area is provided with the following functions.

(a) Registering a watch-expression
CPU registers/categories can be registered in the Watch panel as watch-expressions.
See "2.12.6.1 Register a watch-expression" for details on how to operate it.

Remark 1. When you have registered a watch-expression with a category as the object, all of the CPU registers belonging to that category are registered as watch-expressions.

Remark 2. A scope specification is automatically added to a registered watch-expression.

(2) [Value] area
The values of each CPU register are displayed and changed.
The radix of a data value can be selected by the button on the toolbar or the context menu item. In addition, a display format adding the value in hexadecimal number constantly can also be selected as well.
The meanings of the colors of the CPU register values are as follows (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box):

Display Example (Default)			Description
0x0	Character color	Blue	The value of the CPU register that the user is changing Press the [Enter] key to write to the target memory.
	Background color	Standard color	
0x0	Character color	Brown	The value of the CPU register that has been changed because of the execution of a program. The highlighting is rest by executing again the program.
	Background color	Cream	

This area is provided with the following functions.

(a) Changing the CPU register value
To edit the CPU register value, select the value to edit, then change the value directly from the keyboard after clicking again on it (press the [Esc] key to cancel the edit mode).
After you edit the value of the CPU register, it is written to the target memory of the debug tool by pressing the [Enter] key or moving the focus to outside the edit region.

(b) Saving the contents of the CPU register
The Save As dialog box can be opened by selecting the [File] menu >> [Save CPU Register Data As…], and all the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.12.2.4 Save the CPU register contents" for details on the method for saving the contents of the CPU register.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel. This item is disabled during execution of a program.
Notation	The following buttons to change the notation of a data value are displayed.
AutoSelect	Displays the value of the selected item (including sub-items) in the default notation (default).
Hexadecimal	Displays the value of the selected item (including sub-items) in hexadecimal number.
Signed Decimal	Displays the value of the selected item (including sub-items) in signed decimal number.
Unsigned Decimal	Displays the value of the selected item (including sub-items) in unsigned decimal number.
Octal	Displays the value of the selected item (including sub-items) in octal number.
Binary	Displays the value of the selected item (including sub-items) in binary number.
ASCII	Displays the character string of the selected item (including sub-items) in ASCII code. If the character size is 2 bytes and above, it is displayed with the characters for each 1 byte arranged side-by-side.
Float	Displays the value of the selected item in float. Note that when the value is not 4-byte data, displays it in the default notation.
Double	Displays the value of the selected item in double. Note that when the value is not 8-byte data, displays it in the default notation.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value.

[[File] menu (CPU Register panel-dedicated items)]

The following items are exclusive for the [File] menu in the CPU Register panel (other items are common to all the panels).

Note that all these items are disabled during execution of a program.

Save CPU Register Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the CPU register"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save CPU Register Data As...].
Save CPU Register Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the CPU register").

[[Edit] menu (CPU Register panel-dedicated items)]

The following items are exclusive for [Edit] menu in the CPU Register panel (all other items are disabled).

Cut	Deletes the selected character string and copies it to the clipboard. This item becomes valid only when the character string is being edited.
Copy	Copies the selected character string to the clipboard during editing. If a line is selected, copies the register or the category to the clipboard. The copied item can be pasted to the Watch panel.
Paste	Pasts the character string copied in the clipboard to the caret position. This item becomes valid only when the character string is being edited.
Select All	Selects all the items of this panel.
Find...	Opens the Find and Replace dialog box with selecting the [Find in Files] tab.
Replace...	Opens the Find and Replace dialog box with selecting the [Replace in Files] tab.

[Context menu]

Register to Watch1	Registers the selected register or category to the Watch panel (Watch1).
Copy	Copies the selected character string to the clipboard during editing. If a line is selected, copies the register or the category to the clipboard. The copied item can be pasted to the Watch panel.
Notation	The following cascade menus to specify the notation of a data value are displayed.
AutoSelect	Displays the value of the selected item (including sub-items) in the default notation (default).
Hexadecimal	Displays the value of the selected item (including sub-items) in hexadecimal number.
Signed Decimal	Displays the value of the selected item (including sub-items) in signed decimal number.
Unsigned Decimal	Displays the value of the selected item (including sub-items) in unsigned decimal number.
Octal	Displays the value of the selected item (including sub-items) in octal number.
Binary	Displays the value of the selected item (including sub-items) in binary number.
ASCII	Displays the character string of the selected item (including sub-items) in ASCII code. If the character size is 2 bytes and above, it is displayed with the characters for each 1 byte arranged side-by-side.
Float	Displays the value of the selected item in float. Note that when the value is not 4-byte data, displays it in the default notation.
Double	Displays the value of the selected item in double. Note that when the value is not 8-byte data, displays it in the default notation.
Include Hexadecimal Value	Adds the value in hexadecimal number enclosing with "()" at the end of the value.

IOR panel

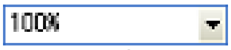
This panel is used to display the contents of the I/O register and change the I/O register values (see "2.12.3 Display/change the I/O register").

This panel appears only when connected to the debug tool.

Caution 1. The I/O registers that cause the microcontroller to operate when they are read are read-protected and therefore cannot be read ("?" is displayed in [Value]).
To read out the value of a read-protected I/O register, select [Force Read Value] from the context menu.
Reading of each register is allowed only once. After [Force Read Value] is applied, the register is no longer marked "?" so it will not be instantly recognizable as read-protected.

Caution 2. When the selected microcontroller supports multi-core, this panel displays/changes the value regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

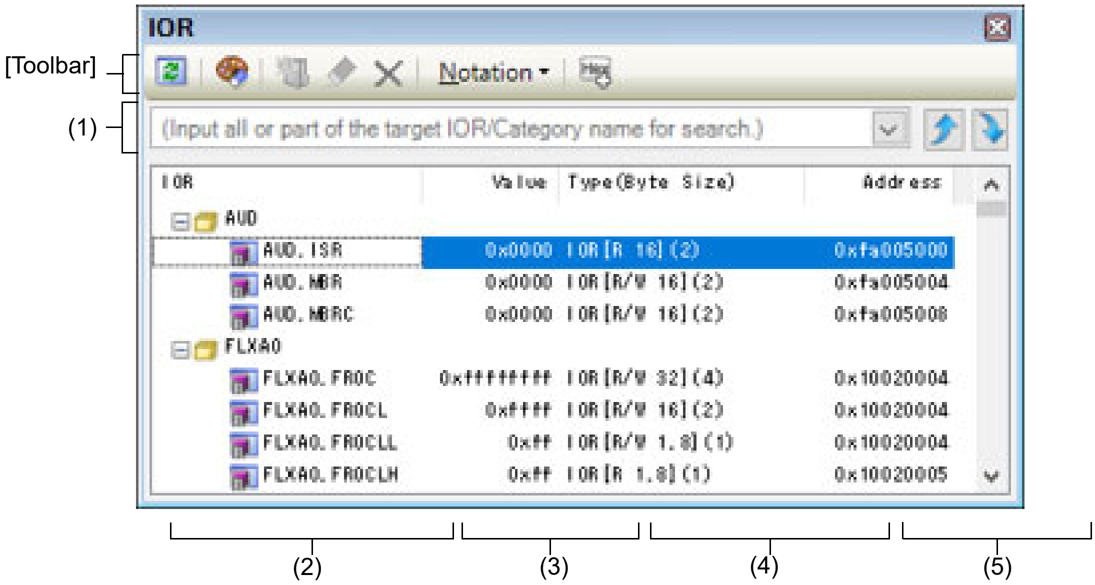
Caution 3. While display of the contents of I/O registers is supported, the display of individual bits is not. The values of individual bits in I/O registers can be monitored by registering the bits in the Watch panel.

Remark 1. This panel can be zoomed in and out by 

in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Remark 2. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A.12 IOR Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (IOR panel-dedicated items)]
- [[Edit] menu (IOR panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [IOR].

[Description of each area]

(1) Search area

This area is used to search for the I/O register name.

	Specifies the character strings to search (case-insensitive). You can either type character strings directly from the key board (up to 512 characters), or select one from the input history via the drop-down list (up to 10 items).
	Searches up for the I/O register name containing the string specified in the text box, and selects the I/O register that is found.
	Searches down for the I/O register name containing the string specified in the text box, and selects the I/O register that is found.

Remark 1. A hidden I/O register name being classified with a category can be searched (the category is opened and the I/O register is selected).

Remark 2. After typing character strings to search, to press the [Enter] key is the same function as clicking the button, and to press the [Shift] + [Enter] key is the same function as clicking the button.

(2) [IOR] area

The types of I/O register are classified as categories (folders), and a list of the respective I/O register name is displayed.

The meanings of the icons are as follows:

	Indicates that the I/O register name belonging to this category is displayed. When you double-click on the icon, or click on the "-" mark, the category is closed and the I/O register name is hidden. Note that no categories exist by default. Perform Tree editing if you need a category.
	Indicates that the I/O register name belonging to this category is hidden. When you double-click on the icon, or click on the "+" mark, the category is opened and the I/O register name is displayed. Note that no categories exist by default. Perform Tree editing if you need a category.
	Indicates the name of the I/O register.

Remark The category names are sorted in character code order by clicking on the header part of this area (the I/O register names in the category are also similarly sorted).

This area is provided with the following functions.

(a) Tree editing

The each I/O register can be categorized (by folders) and displayed in the tree view.

To create a category, Click the button on the toolbar or select [Create Category] from the context menu after moving the caret to a I/O register name to create a category, and then input a desired name from the keyboard (up to 1024 characters).

To delete a category, select the category then click the button on the toolbar or select [Delete] from the context menu. However, the categories that can be deleted are only the empty categories.

To rename the created category, select the category then do either one of the following.

- Click the name again, then directly rename the category name.
- Select the [Edit] menu >> [Rename], then directly rename the category name.
- Press the [F2] key, then directly rename the category name.

By directly dragging and dropping the I/O register in the created category, each I/O register is displayed in the categorized tree view.

Also, the display order of the categories and the I/O register names (upper or lower position) can be changed easily by drag and drop operation.

To clear and restore an edited category, click on the button on the toolbar or select [Clear customization] from the context menu.

Caution 1. Categories cannot be created within categories.

Caution 2. I/O registers cannot be added or deleted.

(b) Registering a watch-expression
Variable names of C language, CPU registers, I/O registers, and assembler symbols can be registered in the Watch panel as watch-expressions.
See "2.12.6.1 Register a watch-expression" for details on how to operate it.

Remark 1. When you have registered a watch-expression with a category as the object, all of I/O registers belonging to that category are registered as watch-expressions.

Remark 2. A scope specification is automatically added to a registered watch-expression.

(3) [Value] area
The value of I/O register is displayed and changed.
The radix of a data value can be selected by the button on the toolbar or the context menu item. In addition, a display format adding the value in hexadecimal number constantly can also be selected as well.
The meanings of the marks and colors displayed as I/O register values are as follows (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box):

Display Example (Default)			Description
0x0	Character color	Blue	The value of the I/O register that the user is changing (press the [Enter] key to write to the target memory).
	Background color	Standard color	
0x0	Character color	Brown	The value of the I/O register that has been changed because of the execution of a program To reset the highlighting, select the button on the toolbar or [Reset Color] from the context menu.
	Background color	Cream	
?	Character color	Gray	The value of the I/O register that is a read-protected object[Note]
	Background color	Standard color	

Note An I/O register for which the microcontroller ends up being activated by a read operation is shown. To read the value of read-protected I/O register, select [Force Read Value] from the context menu.

Caution The timing for acquiring the values differs in the case of a 1 byte/2 bytes I/O register and that of 1 bit I/O registers that have been allocated to a 1 byte/2 bytes I/O register. Owing to this, there are also cases where the values differ even if the value of the same I/O register is displayed.

Remark The values are sorted in ascending order of the numerical values by clicking on the header part of this area.

This area is provided with the following functions.

(a) Changing I/O register values
To edit the I/O register value, select the value to edit, then change the value directly from the keyboard after clicking again on it (press the [Esc] key to cancel the edit mode).
After you edit the value of the I/O register, it is written to the register of the debug tool by pressing the [Enter] key, or moving the focus to outside the edit region.
See "2.12.3.4 Modify the I/O register contents" for details on the method for changing the I/O register value.

(b) Saving the contents of the I/O register
The Save As dialog box can be opened by selecting the [File] menu >> [Save *IOR* Data As...], and all the contents of the I/O register can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.12.3.6 Save the I/O register contents" for details on the method for saving the contents of the I/O register.

(4) [Type (Byte Size)] area
The type information of each I/O register is displayed in the form shown below.

- *<Type of I/O register>* [*<Access attribute>* *<Accessible sizes>*](*<Size>*)

Access attribute	One of the following is displayed as the access attribute.	
	R	Read only
	W	Write only
	R/W	Read/Write

Accessible sizes	All accessible sizes are demarcated by a comma and listed in order of the smallest size in bit units (1 to 32 bits).
Size	The size of the I/O register is displayed. It is displayed by supplying the unit, in byte units in the event that it can be displayed in byte units, and in bit units in the event that it can be displayed on in bit units.

Example 1. "The case of "IOR [R/W 1.8] (1 byte)"
An I/O register that is readable/writable and 1 bit accessible/8 bit accessible, and whose size is 1 byte

Example 2. "The case of "IOR [R/W 1] (1 bit)"
An I/O register that is readable/writable and 1 bit accessible, and whose size is 1 byte

Remark The type information is sorted in the character code order by clicking on the header part of this area.

(5) [Address] area
The address that each I/O register is mapped is displayed (hexadecimal number notation fixing).
However, in the case of the bit register, it is displayed by providing a bit offset value like the following examples.

Example 1. The case of "0xFF40"
This is allocated to the address "OxFF40"

Example 2. The case of "0xFF40.4"
This is allocated to bit 4 of the address "0xFF40.4" (bit register)

Remark The addresses are sorted in ascending order of numerical values by clicking on the header part of this area.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel. Note that the values of read-protected I/O register are not re-read. This item is disabled during execution of a program.
	Resets highlighting of the selected I/O register whose value has been changed by executing a program. Note that this item is disabled during execution of a program.
	Adds a new category (folder). Directly input the category name in the text box. There are no restrictions on the number of categories that can be created anew (however, it is not possible to create a category inside a category). Note that this item is disabled during execution of a program.
	Clears the user customization (creation of categories, changing orders, and changing notation).
	Deletes the selected character string(s). If an empty category is in a select state, its category is deleted (it is not possible to delete I/O registers).
Notation	The following buttons to change the notation of a data value are displayed.
Hexadecimal	Displays the value of the selected item in hexadecimal number (default).
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned Decimal	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.

	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item.

[[File] menu (IOR panel-dedicated items)]

The following items are exclusive for the [File] menu in the IOR panel (other items are common to all the panels). Note that all these items are disabled during execution of a program.

Save IOR Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the I/O register"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save IOR Data As...].
Save IOR Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the I/O register").

[[Edit] menu (IOR panel-dedicated items)]

The following items are exclusive for [Edit] menu in the IOR panel (all other items are disabled).

Cut	Deletes the selected character string(s) and copies them to the clipboard (it is not possible to cut I/O registers/categories).
Copy	Copies the contents of the selected range to the clipboard as character string(s). If the I/O register(s)/category(s) are selected, copies them to the clipboard. The copied item can be pasted to the Watch panel.
Paste	If texts are in editing, pastes the contents of the clipboard to the caret position (it is not possible to paste I/O registers/categories).
Delete	Deletes the selected character string(s). If an empty category is in a select state, its category is deleted (it is not possible to delete I/O registers).
Select All	If texts are in editing, selects all the character strings. If texts are not in editing, selects all the I/O registers/categories.
Rename	Edits the name of the selected category.
Find...	Moves the focus to the text box in the Search area.
Move...	Opens the Go to the Location dialog box to move the caret to the specified I/O register.

[Context menu]

Register to Watch1	Registers the selected I/O register or category to the Watch panel (Watch1).
Refresh	Acquires the latest data from the debug tool, and updates the contents of this panel. Note that the values of read-protected I/O register are not re-read. This item is disabled during execution of a program.
Force Read Value	Forcibly reads once the value of the read-protected I/O register.
Move...	Opens the Go to the Location dialog box.
Create Category	Adds a new category (folder). Directly input the category name in the text box. There are no restrictions on the number of categories that can be created anew (however, it is not possible to create a category inside a category). Note that this item is disabled during execution of a program.
Clear customization	Clears the user customization (creation of categories, changing orders, and changing notation).

Copy	Copies the contents of the selected range to the clipboard as character string(s). If the I/O register(s)/category(s) are selected, copies them to the clipboard. The copied item can be pasted to the Watch panel.
Delete	Deletes the selected character string(s). If an empty category is in a select state, its category is deleted (it is not possible to delete I/O registers).
Notation	The following cascade menus are displayed to specify the notation.
Hexadecimal number	Displays the value of the selected item in hexadecimal number (default).
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned decimal number	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.
Include Hexadecimal Value	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item.
Reset Color	Resets highlighting of the selected I/O register whose value has been changed by executing a program.

Local Variables panel

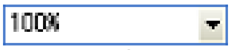
This panel is used to display the contents of the local variable and change the local variable values (see "2.12.5 Display/change local variables").

This panel appears only when connected to the debug tool.

Caution 1. Nothing is displayed on this panel during execution of a program. When the execution of a program is stopped, items in each area are displayed.

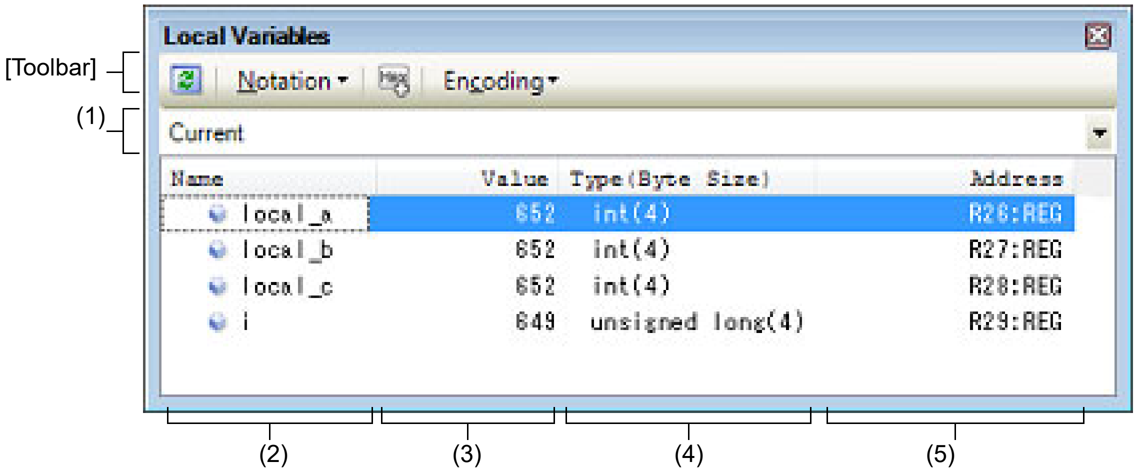
Caution 2. Due to compiler optimization, the data for the target variable may not be on the stack or in a register in blocks where that variable is not used. In this case, the target variable will not be displayed.

Caution 3. When the selected microcontroller supports multi-core, this panel displays/changes the value regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

Remark 1. This panel can be zoomed in and out by  in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Note 2. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A.13 Local Variables Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Local Variables panel-dedicated items)]
- [[Edit] menu (Local Variables panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Local Variable].

[Description of each area]

(1) Scope area

Select the scope of the local variable to be displayed from the following drop-down list.

Item	Operation
Current	Displays local variables in the scope of the current PC value.

Item	Operation
<Depth> <Function name() [*file name#line number*]*>*[Note]	Displays local variables in the scope of the calling function. After the program is executed, the scope that is selected is maintained as long as the selected scope exists.

Note The calling functions displayed in the Call Stack panel are displayed.

(2) [Name] area

The local variable name or function name is displayed.
The argument of the function is also displayed as the local variable.
In addition, the hierarchical structure is displayed in tree format for arrays, pointer variables, and structures or unions.
This area cannot be edited.
The meanings of the icons are as follows:

	Indicates the variable. Auto variables, internal static variables, and register variables are also displayed[Note]. In addition, the hierarchical structure is displayed in tree format for arrays, pointer variables, and structures or unions. If "+" mark exist at the top of the name, the next structure is expanded by clicking it (the mark changes to "-" after the expansion).	
	Array	All elements in the array
	Pointer variables	Variables that the pointer designates If the pointer designates a pointer, add "+" mark and expand it by clicking the mark. Note that if the pointer designates an unknown, "?" mark is displayed.
	Structures/Unions	All the member of structures/unions
	Indicates the argument.	
	Indicates the function.	

Note When Auto variables are used to display local variables, accurate values cannot be displayed at a prologue ("{") or epilogue ("}") of a function. The Auto variable addresses are the relative addresses from the address pointed to by the stack pointer (SP), so their addresses are not determined until the SP value is determined in the function. The SP is manipulated via prologues or epilogues, so the accurate value cannot be displayed.

This area is provided with the following functions.

(a) Registering a watch-expression

Variable names of C language can be registered in the Watch panel as watch-expressions.
See "2.12.6.1 Register a watch-expression" for details on how to operate it.

Remark A scope specification is automatically added to a registered watch-expression.

(b) Jump to memory

By selecting [Jump to Memory] from the context menu, the Memory panel (Memory1) opens with moving the caret to the source line corresponding to the address where the selected local variable is disposed (if the Memory panel (Memory1) is already open, the screen will jump to the panel).

(3) [Value] area

The value of the local variable is displayed and changed.
The notation of a data value can be selected by the button on the toolbar or the context menu item. In addition, a display format adding the value in hexadecimal number constantly can also be selected as well.
The meanings of the marks and colors displayed as the values of the local variables are as follows (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box):

Display Example (Default)			Description
0x0	Character color	Blue	The value of the local variable that the user is changing Press the [Enter] key to write to the target memory.
	Background color	Standard color	
0x0	Character color	Brown	The value of the local variable that have been changed because of the execution of a program[Note]. The highlighting is rest by executing again the program.
	Background color	Cream	
?	Character color	Gray	The value of the local variable that could not be acquired.
	Background color	Standard color	

Note Variables that the name stays same from the start point where the program started executing to the breakpoint and their values are changed are the target.

This area is provided with the following functions.

(a) Changing the local variable/argument value
To edit the local variable value or the argument value, select the value to edit, then change the value directly from the keyboard after clicking again on it (press the [Esc] key to cancel the edit mode).
After you edit the value of the local variable or the argument, it is written to the target memory of the debug tool by pressing the [Enter] key or moving the focus to outside the edit region.
See "2.12.5.2 Modify the contents of local variables" for details on the method for changing the local variable/argument value.

(b) Saving the contents of the local variable
The Save As dialog box can be opened by selecting the [File] menu >> [Save Local Variables Data As...], and all the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.12.5.3 Save the contents of local variables" for details on the method for saving the contents of the local variable.

(4) [Type (Byte Size)] area
The type name of the local variable is displayed. The notation accords with the description of C language.
For an array, an element number is displayed in "[]". For a function, its size (number of bytes) is displayed in "()".
This area cannot be edited.

(5) [Address] area
The address of the local variable is displayed. When a variable is assigned to the register, the name of the register is displayed.
This area cannot be edited.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel.
Notation	The following buttons to specify the notation of values are displayed.
AutoSelect	Displays values on this panel in the default notation according to the type of variable (default).
Hexadecimal	Displays values on this panel in hexadecimal number.
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.
Decimal Notation for Array Index	Displays array indexes on this panel in decimal number (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel in hexadecimal number.
Float	Displays values on this panel in float. Note that when the value is not 4-byte data, or has the type information, displays it in the default notation.
Double	Displays values on this panel in double. Note that when the value is not 4-byte data, or has the type information, displays it in the default notation.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value.
Encoding	The following buttons to specify the encoding of character variables are displayed.
	Displays character variables in ASCII code (default).
	Displays character variables in Shift-JIS code.
	Displays character variables in EUC-JP code.
	Displays character variables in UTF-8 code.
	Displays character variables in UTF-16 Big-Endiancode.
	Displays character variables in UTF-16 Little-Endian code.
	Displays character variables in UTF-32 Big-Endian code.
	Displays character variables in UTF-32 Little-Endiancode.

[[File] menu (Local Variables panel-dedicated items)]

The following items are exclusive for the [File] menu in the Local Variables panel (other items are common to all the panels).

Note that all these items are disabled during execution of a program.

Save Local Variables Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the local variable"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Local Variables Data As...].
Save Local Variables Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of the local variable").

[[Edit] menu (Local Variables panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Local Variables panel (all other items are disabled).

Copy	Copies the contents of the selected line or the character string to the clipboard.
Select All	Selects all the items of this panel.
Rename	Changes to the edit mode to edit the selected local variable value (see "2.12.5.2 Modify the contents of local variables"). This item is disabled during execution of a program.
Find...	Opens the Find and Replace dialog box with selecting the [Find in Files] tab.
Replace...	Opens the Find and Replace dialog box with selecting the [Replace in Files] tab.

[Context menu]

Register to Watch1		Registers the selected local variable to the Watch panel (Watch1).
Copy		Copies the contents of the selected line or the character string to the clipboard.
Notation		The following cascade menus to specify the notation of values are displayed.
	AutoSelect	Displays values on this panel in the default notation according to the type of variable (default).
	Hexadecimal	Displays values on this panel in hexadecimal number.
	Decimal	Displays values on this panel in decimal number.
	Octal	Displays values on this panel in octal number.
	Binary	Displays values on this panel in binary number.
	Decimal Notation for Array Index	Displays array indexes on this panel in decimal number (default).
	Hexadecimal Notation for Array Index	Displays array indexes on this panel in hexadecimal number.
	Float	Displays values on this panel in float. Note that when the value is not 4-byte data, or has the type information, displays it in the default notation.
	Double	Displays values on this panel in double. Note that when the value is not 4-byte data, or has the type information, displays it in the default notation.
Include Hexadecimal Value		Adds the value in hexadecimal number enclosing with "()" at the end of the value.
Encoding		The following cascade menus to specify the encoding of character variables are displayed.

ASCII	Displays character variables in ASCII code (default).
Shift_JIS	Displays character variables in Shift-JIS code.
EUC-JP	Displays character variables in EUC-JP code.
UTF-8	Displays character variables in UTF-8 code.
UTF-16 Big-Endian	Displays character variables in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character variables in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character variables in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character variables in UTF-32 Little-Endian code.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the memory value corresponding to the address of the selected line in this panel.

Watch panel

This panel is used to display the contents of the registered watch-expressions and change their values (see "2.12.6 Display/change watch-expressions").

Up to a maximum of four of these panels can be opened. Each panel is identified by the names "Watch1", "Watch2", "Watch3", and "Watch4" on the titlebar, and the watch-expressions can be registered/deleted/moved individually.

Watch-expressions can be registered in this panel as well as in the Editor panel, Disassemble panel, Memory panel, CPU Register panel, Local Variables panel or IOR panel.

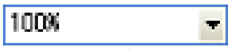
When the panel is closed with registered watch-expressions, the panel closes but the information on the registered watch-expressions is retained. Therefore, if the same panel is opened again, it is opened with the watch-expressions registered.

The display contents are automatically updated when the value of the watch-expression changes after a program is executed (when the execution is done in steps, the display is updated after each step).

In addition, by enabling the Real-time display update function, it is also possible to update the display contents in real-time even while a program is being executed.

This panel appears only when connected to the debug tool.

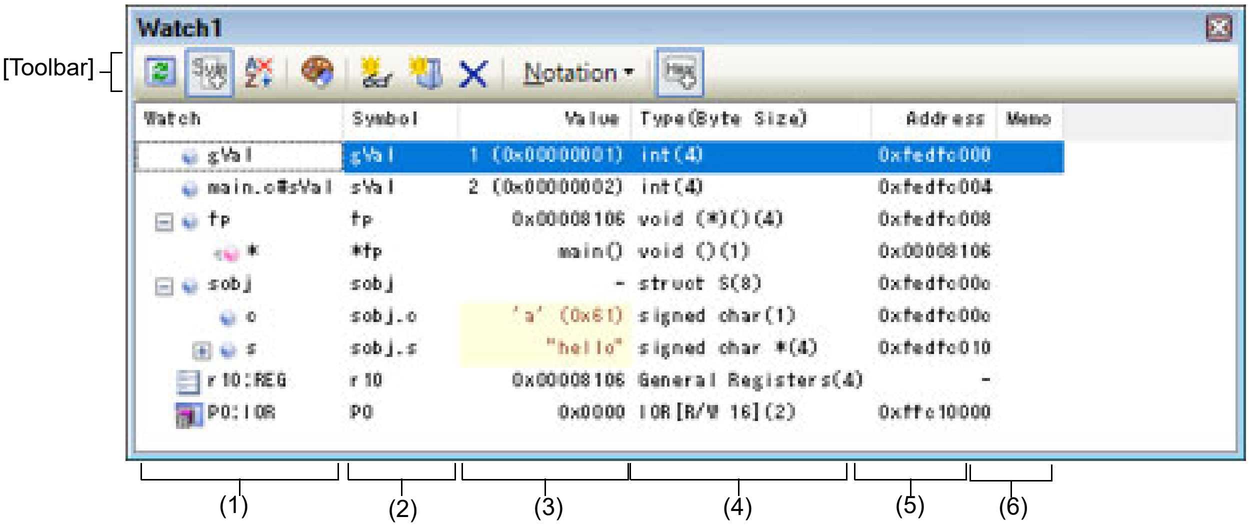
Caution When the selected microcontroller supports multi-core, this panel displays/changes the value regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

Remark 1. This panel can be zoomed in and out by 

in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Remark 2. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A.14 Watch Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Watch panel-dedicated items)]
- [[Edit] menu (Watch panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Watch] >> [Watch*1 - 4*].

[Description of each area]

(1) [Watch] area
All the registered watch-expressions are displayed in a list.
Clicking the title of the list in this area sorts the watch-expressions in the list in alphabetical order. Categories (folders) can be created to categorize the watch-expressions and display them in the tree view (see "(a) Tree editing").
The meanings of the icons are as follows:

	Indicates that the watch-expression belonging to this category is displayed. When you double-click on the icon, or click on the "-" mark, the category is closed and the watch-expression is hidden.
	Indicates that the watch-expression belonging to this category is hidden. When you double-click on the icon, or click on the "+" mark, the category is opened and the watch-expression is displayed.
	Indicates that the watch-expression is a variable. At the top of the watch-expression represents arrays, pointer type variables, and structures/unions, "+"/"-" mark is displayed. Click the mark to Expand/shrink display.
	Indicates that the watch-expression is a function.
	Indicates that the watch-expression is an immediate value.
	Indicates that the watch-expression is an expression.
	Indicates that the watch-expression is I/O register.
	Indicates that the watch-expression is CPU register. At the top of the watch-expression that has the lower level register (part of the register), "+"/"-" mark is displayed. Click the mark to Expand/shrink display.

This area is provided with the following functions.

(a) Tree editing
Watch-expressions can be categorized (by folders) and displayed in the tree view.
To create a category, click the  button on the toolbar or select [Create Category] from the context menu after moving the caret to the position to create a category, and then input a desired name from the keyboard.
To delete a category, select the category then click the  button on the toolbar or select [Delete] from the context menu.
To rename the created category, select the category then do either one of the following.

- Click the name again, then directly rename the category name.
- Select the [Edit] menu >> [Rename], then directly rename the category name.
- Press the [F2] key, then directly rename the category name.

By directly dragging and dropping the registered watch-expression in the created category, each category is displayed in the categorized tree view. Also, the display order of the categories and the watch-expressions (upper or lower position) can be changed easily by drag and drop operation.

Caution 1. Categories cannot be created within categories.

Caution 2. Up to 1500 categories can be created in one watch panel (if this restriction is violated, a message appears).

Remark Drag and drop the watch-expressions/categories in other watch panel (Watch1 to Watch4) to copy them.

(b) Expand/shrink display
At the top of the watch-expression represents arrays, pointer type variables, structures/unions, and registers (with the name of the part), "+"/"-" mark is displayed. Click the mark to expand the contents ("+" mark is changed to "-" after the expansion).

Watch-Expression	Contents When Expanded
Array	All elements in the array Select [Encoding] >> [ASCII] from the context menu to display the value as a string (up to 256 characters). Note, however, that any characters that cannot be displayed in the encoding will be shown as periods "." or "?".
Pointer type variable	Variables that the pointer designates
Structure/Union	All the member of structure/union
Register	Name of the bit/bit string that constructs register Example) ECR register FECC register EICC register

(c) Registering new watch-expression

There are following three methods of registering a new watch expression.

<1> Register from other panels

Do either one of the following to register watch-expressions in other panels.

- Drag and drop the target character string onto this area in the desired watch panel (Watch1 to Watch4).
- Select [Register to Watch1] from the context menu after selecting the target character string or place the caret on either of the target character string (the target is automatically determined).
- Select the [Edit] menu >> [Paste] in this area in the desired watch panel (Watch1 to Watch4) after selecting the [Edit] menu >> [Copy] for the target character string.

The relationship between panels that can use this operation and targets that can be registered as watch-expressions is as follows:

Table A.2 Relationship between Panels and Targets That Can be Registered as Watch-Expressions

Panel Name	Targets That can be Registered as Watch-Expressions
Editor panel	Variable names of C language, CPU registers, I/O registers, and assembler symbols
Disassemble panel	Variable names of C language, CPU registers, I/O registers, and assembler symbols
CPU Register panel	CPU registers[Note]
Local Variables panel	Variable names of C language (local variables)
IOR panel	I/O registers[Note]

Note The scope-specification is automatically added to the registered watch-expression.

<2> Directly register in the Watch panel

Click the  button on the toolbar or select [Add New Watch] from the context menu in the desired watch panel (Watch1 to Watch4) to display an entry box for a new watch-expression in the bottom of this area.
Directly input a watch-expression from the keyboard in the [Watch] area in the entry box then press the [Enter] key.
For details on the input format of the watch-expression, see "(b) Watch-expression and operator".

Watch-expressions can be registered with specifying the scope. The scope specifications with watch-expression registration are as follows:

Caution 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

Caution 2. If functions with the same name exist, write the type of parameter expressly. Example: func(int, int)

Table A.3 Handling of a C Language Function When Registered in Watch by Specifying Scope

Scope Specification	Load Module File Name	Source File Name	Function Name	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]

Note 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.

Table A.4 Handling of a C Language Variable When Registered in Watch by Specifying Scope

Scope Specification	Load Module File Name	Source File Name	Function Name	Variable Name	Subject to be searched
prog$file#func#var	prog	file	func	var	Static variables inside a static function[Note 1]
prog$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 1]
file#var	Current	file	Global	var	Static variables inside a file
var	Current	Current	Current	var	All[Note 2]

Note 1. 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 2. 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. Note, if [Yes] is selected for the [Search for global variables even outside the current scope] property, searching for global variables that are outside the current scope will also proceed.

Table A.5 Handling of a CPU Register When Registered in Watch by Specifying Scope

Scope Specification	System Register	Name of CPU Register
r10:REG	(None)	r10
name:SR0102[Note]	regID1, selID2	name

Note Specify regID and selID as decimal values to register the system registers. For values of regID and selID, refer to the hardware manual for the device. In this table, specify the desired character string for "name" in [Name of CPU Register].

Table A.6 Handling of an I/O Register when Registered in Watch by Specifying Scope

Scope Specification	Name of I/O register
P0:IOR	P0
P0	P0

Remark 1. A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this area (see "2.21.2 Symbol name completion function").

Remark 2. An immediate value is treated as an address. Note, however, that an immediate value with operators cannot be used.

Remark 3. An arithmetic expression with symbols cannot be used for a watch-expression.

Remark 4. If the same name exists either in C language variables, CPU registers or I/O registers, and it is registered without specifying scopes, then its value will be displayed after the symbol is determined in the following order.
Variable of C language > CPU registers > I/O register
If "$" is specified at the top of a watch-expression, then its value will be displayed after the symbol is determined in the following order.
CPU registers > I/O register > Variable of C language

Remark 5. If a local variable and a global variable exist with the same name, and its symbol name is registered without specifying scopes, then its value will be displayed after the symbol is determined based on the scope of the current PC value.

Remark 6. If the letter "I" alone is specified as a watch-expression, it is interpreted as an imaginary keyword. To acquire the value of a register "I," add ":REG" after the register.

Remark 7. When watch-expressions are registered from the IOR panel or the CPU Register panel, the scope specification is automatically added.

<3> Register from other application
Select a character string of a variable of C language, CPU register, I/O register or assembler symbol from a external editor then do either one of the following.

- Drag and drop the target character string in this area in the desired watch panel (Watch1 to Watch4).
- Select the [Edit] menu >> [Paste] in this area in the desired watch panel (Watch1 to Watch4) after copying the target character string.

Caution 1. Up to 3000 watch-expressions can be registered in one watch panel (if this restriction is violated, a message appears).

Caution 2. Due to compiler optimization, the data for the target variable may not be on the stack or in a register in blocks where that variable is not used. In this case, the target watch-expression value is displayed as "?".

Remark 1. Each watch-expression registered in each watch panel (Watch1 to Watch4) is managed in each panel and saved as the user information of the project.

Remark 2. More than one watch-expression with the same name can be registered.

Remark 3. You can export registered watch-expressions to a file and import it so that the watch-expressions can be re-registered (see "2.12.6.8 Export/import watch-expressions").

(d) Editing watch-expression
To edit the registered watch-expression, double-click the watch-expression to be edited to change the watch-expression to edit mode then directly edit from the keyboard (press the [Esc] key to cancel the edit mode).
After editing the watch-expression, press the [Enter] key to complete the editing.

(e) Deleting watch-expression
To delete the registered watch-expression, select the watch-expression(s) to be deleted then click the  button on the toolbar or select [Delete] from the context menu.

(f) Setting of various events
Various events can be set to the selected watch-expression by selecting [Access Break], [Trace Output] or [Performance Measurement] from the context menu.
If an access event is set, the mark of the watch-expression is changed (the event mark of a break event, Trace event or Performance Measurement event is displayed under the icon of the watch-expression in layers).
When an event is set, the detailed information about the set event is reflected in the Events panel.
Note that events are only set to the watch-expressions that are global variables, static variables inside functions, or file-internal static variables.
See the following for details on how to set events.

- "2.11.5 Stop the program with the access to variables/I/O registers"
- "2.14.4 Collect execution history only when the condition is met"
- "2.16.1 Measure the performance in a section"

(g) Jump to the address with memory definition

By selecting [Jump to Memory] from the context menu, the Memory panel (Memory1) opens with moving the caret to the address in which the selected watch-expression is defined (if the Memory panel (Memory1) is already open, the screen will jump to the panel).
Note that this operation is disabled when more than one watch-expression is selected at the same time or the CPU register/I/O register is selected.

(2) [Symbol] area
Symbols that have been registered as watch-expressions are displayed.
The display of the [Symbol] area is switched by selecting [Displays symbol column] from the context menu or toolbar.
When watch-expressions have a specified scope, the [Symbol] area is displayed without specifying the scopes of watch-expressions.

Example var for prog$file#func#var

When arrays, pointer-type variables, and structures, unions, or classes of the watch-expression are expanded, the [Symbol] area displays watch-expressions including the name of the parent node.

Example sobj.mem for the mem member expanded from the sobj structure object

(3) [Value] area
The value of the registered watch-expression is displayed and changed (if the watch-expression is a function pointer, the function name is displayed in this area).
Notations and encodes can be selected by the button on the toolbar or the context menu item. In addition, a display format adding the value in hexadecimal number constantly can also be selected as well.
The default display format of the values is automatically decided depending on the type of the watch-expression.

Table A.7 Display Format of Watch-Expressions (Default)

Type of Watch-Expression	Display Format
char, signed char, unsigned char	ASCII code with hexadecimal number
short, signed short, short int, signed short int, int, signed, signed int, long, signed long, long int, signed long int	Signed decimal number with hexadecimal number
unsigned short, unsigned short int, unsigned, unsigned int, unsigned long, unsigned long int	Unsigned decimal number with hexadecimal number
float	Float (when the size is 4-byte) with hexadecimal number
double, long double	Double (when the size is 8-byte) with hexadecimal number
Pointers to char, signed char, unsigned char	Characters Encoding: ASCII
Pointers to other than char, signed char, unsigned char	Hexadecimal number
Arrays of char, signed char, unsigned char types	Characters Encoding: ASCII
bit, boolean, _boolean	Unsigned decimal number with hexadecimal number
Enumeration type	Enumeration constant value with hexadecimal number
Label, address of immediate value, EQU symbol	Signed decimal number with hexadecimal number
bit symbol	Unsigned decimal number with hexadecimal number
Others	Hexadecimal number

The meanings of the marks and colors displayed as the values of watch-expressions are as follows (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box):

Display Example (Default)			Description
0x0	Character color	Blue	The value of the watch-expression that the user is changing Press the [Enter] key to write to the target memory.
	Background color	Standard color	
0x0	Character color	Pink	The value of the watch-expression that is displayed with the Real-time display update function
	Background color	Standard color	
0x0	Character color	Brown	The value of the watch-expression that has been changed because of the execution of a program To reset the highlighting, select the button on the toolbar or [Reset Color] from the context menu.
	Background color	Cream	
?	Character color	Gray	Variable that does not exist is registered as a watch-expression or the value of the watch-expression cannot be retrieved (i.e. when the I/O register is read-protected[Note], or a variable is out of the scope, etc.)
	Background color	Standard color	

Note The I/O register that cause the microcontroller to operate when it is read is read-protected and therefore cannot be read ("?" is displayed in the value).
To read out the value of a read-protected I/O register, select [Force Read Value] from the context menu. Reading of each register is allowed only once.

Remark 1. Each watch-expression acquires the value in the order it was registered.
As the timing to acquire a value is different, the values displayed may be different if the same I/O register is registered more than once.

Remark 2. When a hexadecimal value is also given, then values in the specified notation and hexadecimal values are read separately. For this reason, the values with the specified notion and the hexadecimal values may differ due to the time lag between being read.

This area is provided with the following functions.

(a) Real-time display update function
Using the real-time display update function allows you to display/modify the value of the watch-expression not only while the program is stopped, but also in execution.
See "2.12.1.4 Display/modify the memory contents during program execution" for details on the real-time display update function.

(b) Changing values of watch-expressions
To edit the value of the watch-expression, change the value directly from the keyboard after double-clicking on the value to be edited (press the [Esc] key to cancel the edit mode).
After you edit the value of the watch-expression, it is written to the target memory of the debug tool by pressing the [Enter] key, or moving the focus to outside the edit region.
See "2.12.6.6 Modify the contents of watch-expressions" for detail on how to change values of watch-expressions.

(c) Saving the contents of watch-expressions
By selecting the [File] menu >> [Save Watch Data As...], the Save As dialog box can be opened, and all the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
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).
See "2.12.6.9 Save the contents of watch-expressions" for details on the method for saving the contents of watch-expressions.

(4) [Type (Byte Size)] area
The type information of watch-expressions with the following format is displayed.

Watch-Expression	Display Format
Single CPU register	*<Types of CPU register>* (*<Size*[Note 1]>)

<table>
<tr><th>Watch-Expression</th><th colspan="2">Display Format</th></tr>
<tr><td rowspan="3">Single I/O register</td><td colspan="2"><i><I/O register type></i> (<i><Access attribute> <Accessible sizes><Size</i>[Note 1]>)</td></tr>
<tr><td>Access attribute</td><td>R: Read only
W: Write only
R/W: Read/Write only</td></tr>
<tr><td>Accessible sizes</td><td>All accessible sizes are demarcated by a comma and listed in order of the smallest size in bit units (1 to 32 bits).</td></tr>
<tr><td>Unknown</td><td colspan="2">?</td></tr>
<tr><td>Others</td><td colspan="2"><i><Watch-expression type that follow the C compiler's determination</i>[Note 2]> (<i><Size</i>[Note 1]>)</td></tr>
</table>

Note 1. The size of the watch-expression is displayed in bytes.
However, for bit I/O register or C language bit field, the size is displayed in bits and "bits" is added to the end of the number.

Note 2. Types to be treated are displayed when compiling the watch-expression.

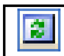
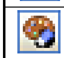
(5) [Address] area
The address that each watch-expression is mapped is displayed (hexadecimal number notation fixing).
If the watch-expression is single CPU register or is unknown, "-" or "?" is displayed instead.

Remark If the watch-expression is the bit I/O register, the bit-offset value is also displayed as follows:

Example When the bit register is allocated to bit 4 of the address "0xFF40":
Display example: 0xFF40.4

(6) [Memo] area
The user can write comments for the watch-expressions/categories.
Each comment for a watch-expression/category written in this area is saved individually as the user information of the project. Therefore, when any of the watch-expression/category is deleted, the comment corresponding to it is also deleted.
Note that when arrays or register are displayed expanded, the comment cannot be input for each element.
To edit the comment, input the character strings directly from the keyboard after double-clicking on the item to be edited (press the [Esc] key to cancel the edit mode). Up to 256 character strings can be input (line feed code is ignored). After editing the character strings, complete the editing by pressing the [Enter] key or moving the focus to outside the edit region.

[Toolbar]

	Reacquires all the values of the registered watch-expression and updates the display. Note that read-protected I/O register values are not re-read.
	Switches between display and non-display of the [Symbol] area.
	Resets highlighting of the selected watch-expression whose value has been changed by executing a program. This item is disabled during execution of a program.
	Registers a new watch-expression. Directly input the watch-expression in the text box (see "(c) Registering new watch-expression") Note that up to 3000 watch-expressions can be registered in one watch panel.
	Adds a new category (folder). Directly input the category name in the text box. Note that up to 1500 categories can be created in one watch panel (categories cannot be created in categories).
	Deletes the selected character string(s). If the watch-expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch-expression is selected).

Notation	The following buttons to change the notation of a data value are displayed.
AutoSelect	Displays the value of the selected watch-expression in the default notation (see "Table A.7 Display Format of Watch-Expressions (Default)") according to the type of variable (default).
Hexadecimal	Displays the value of the selected item in hexadecimal number.
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned Decimal	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.
Float	Displays the value of the selected item in float. Note that this item becomes valid only when the selected watch-expression value is 4-byte data.
Double	Displays the value of the selected item in double. Note that this item becomes valid only when the selected watch-expression value is 8-byte data.
	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item (except the item displayed in hexadecimal number).

[[File] menu (Watch panel-dedicated items)]

The following items are exclusive for the [File] menu in the Watch panel (other items are common to all the panels). Note that all these items are disabled during execution of a program.

Save Watch Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(c) Saving the contents of watch-expressions"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Watch Data As...].
Save Watch Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(c) Saving the contents of watch-expressions").

[[Edit] menu (Watch panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Watch panel (all other items are disabled).

Cut	Deletes the selected character string(s) and copies them to the clipboard. If the watch-expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch-expression is selected).
Copy	Copies the contents of the selected range to the clipboard as character string(s). If the watch-expression(s)/category(s) are selected, copies them to the clipboard (except when the expanded item of the watch-expression is selected).
Paste	If texts are in editing, pastes the contents of the clipboard to the caret position. If texts are not in editing and the watch-expression(s) are copied in the clipboard, registers them to the caret position.
Delete	Deletes the selected character string(s). If the watch-expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch-expression is selected).
Select All	If texts are in editing, selects all the character strings. If texts are not in editing, selects all the watch-expressions/categories.

Rename	Renames the selected watch-expression/category.
Find...	Opens the Find and Replace dialog box with selecting the [Find in Files] tab.
Replace...	Opens the Find and Replace dialog box with selecting the [Replace in Files] tab.

[Context menu]

Access Break	This item becomes valid only when the selected watch-expression is the global variable, the static variable inside functions, the file-internal static variable, or I/O register (multiple selections not allowed). The following cascade menus are displayed to set the access break event (see "2.11.5.1 Set a break event (access type)").
Set Read Break to	Sets a break event with read access condition to the selected watch-expression.
Set Write Break to	Sets a break event with write access condition to the selected watch-expression.
Set R/W Break to	Sets a break event with read/write access condition to the selected watch-expression.
Trace Output	This item becomes valid only when the selected watch-expression is a global variable, static variable inside functions, file-internal static variable, or I/O register (multiple selections not allowed). The following cascade menus are displayed to set the trace-related event (see "2.14.4.1 Set a Point Trace event").
Record Reading Value	Sets a Point Trace event to record the values in the trace memory when the selected watch-expression is accessed for read.
Record Writing Value	Sets a Point Trace event to record the values in the trace memory when the selected watch-expression is accessed for write.
Record R/W Value	Sets a Point Trace event to record the values in the trace memory when the selected watch-expression is accessed for read/write.
Record Start R/W Value [E1][E20]	Sets a trace start event to start collecting the trace data when the selected watch-expression is accessed for read/write.
Record End R/W Value [E1][E20]	Sets a trace end event to stop collecting the trace data when the selected watch-expression is accessed for read/write.
Trace	Opens the Trace panel and displays the acquired trace data.

Performance Measurement Settings [Full-spec emulator] [E1/E20]	This item becomes valid only when the selected watch-expression is a global variable, static variable inside functions, file-internal static variable, or I/O register (multiple selections not allowed). The following cascade menus are displayed to set the performance measurement-related event.
Set Performance Measurement Start Read Value	Sets a performance measurement start event that causes performance measurement to start in response to reading of the selected watch-expression.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Set Performance Measurement End Read Value	Sets a performance measurement end event that causes performance measurement to stop in response to reading of the selected watch-expression.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
Set Performance Measurement Start Write Value	Sets a performance measurement start event that causes performance measurement to start in response to writing of the selected watch-expression.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Set Performance Measurement End Write Value	Sets a performance measurement end event that causes performance measurement to stop in response to writing of the selected watch-expression.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
Set Performance Measurement Start R/W Value	Sets a performance measurement start event that causes performance measurement to start in response to reading/writing of the selected watch-expression.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement start event is set.
Set Performance Measurement End R/W Value	Sets a performance measurement end event that causes performance measurement to stop in response to reading/writing of the selected watch-expression.
Set Performance Measurement *n*	Specify a channel *n* (*n*: 1 to 3) in which a performance measurement end event is set.
Periodic Updating	The following cascade menus are displayed to set for the real-time display update function (see "(a) Real-time display update function").
Periodic Updating Options	Opens the Property panel to set for the real-time display update function.
Refresh	Reacquires all the values of the registered watch-expression and updates the display. Note that the values of read-protected I/O register are not re-read.
Symbol Column	Switches between display and non-display of the [Symbol] area.
Force Read Value	Forcibly reads once the values of the read-protected I/O register. This item is disabled during execution of a program.
Add New Watch	Registers a new watch-expression. Directly input the watch-expression in the text box (see "(c) Registering new watch-expression") Note that up to 3000 watch-expressions can be registered in one watch panel.

Create Category	Adds a new category (folder). Directly input the category name in the text box. Note that up to 1500 categories can be created in one watch panel (categories cannot be created in categories).
Delete	Deletes the selected character string(s). If the watch-expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch-expression is selected).
Cut	Deletes the selected character string(s) and copies them to the clipboard. If the watch-expression(s)/category(s) are selected, deletes them (except when the expanded item of the watch-expression is selected).
Copy	Copies the contents of the selected range to the clipboard as character string(s). If the watch-expression(s)/category(s) are selected, copies them to the clipboard (except when the expanded item of the watch-expression is selected).
Paste	If texts are in editing, pastes the contents of the clipboard to the caret position. If texts are not in editing and the watch-expression(s) are copied in the clipboard, registers them to the caret position.
Rename	Renames the selected watch-expression/category.
Import Watch Expression...	Opens the Open Watch Expression Data File dialog box to import watch-expressions (see "2.12.6.8 Export/import watch-expressions").
Notation	The following cascade menus are displayed to specify the notation.
AutoSelect	Displays the value of the selected watch-expression in the default notation (see "Table A.7 Display Format of Watch-Expressions (Default)") according to the type of variable (default).
Hexadecimal number	Displays the value of the selected item in hexadecimal number.
Signed Decimal	Displays the value of the selected item in signed decimal number.
Unsigned decimal number	Displays the value of the selected item in unsigned decimal number.
Octal	Displays the value of the selected item in octal number.
Binary	Displays the value of the selected item in binary number.
ASCII	Displays the value of the selected item in ASCII code.
Include Hexadecimal Value	Adds the value in hexadecimal number enclosing with "()" at the end of the value of the selected item (except the item displayed in hexadecimal number).
Float	Displays the value of the selected item in float. Note that when the selected watch-expression value is not 4-byte data, or has the type information, displays it in the default notation (see "Table A.7 Display Format of Watch-Expressions (Default)").
Double	Displays the value of the selected item in double. Note that when the selected watch-expression value is not 8-byte data, or has the type information, displays it in the default notation (see "Table A.7 Display Format of Watch-Expressions (Default)").
Decimal Notation for Array Index	Displays array indexes on this panel in decimal number (default).
Hexadecimal Notation for Array Index	Displays array indexes on this panel in hexadecimal number.

Encoding	The following cascade menus are displayed to specify the character code.
ASCII	Displays the value of the selected item in ASCII code (default).
Shift_JIS	Displays the value of the selected item in Shift-JIS code.
EUC-JP	Displays the value of the selected item in EUC-JP code.
UTF-8	Displays the value of the selected item in UTF-8 code.
UTF-16 Big-Endian	Displays the value of the selected item in UTF-16 Big-Endiancode.
UTF-16 Little-Endian	Displays the value of the selected item in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays the value of the selected item in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays the value of the selected item in UTF-32 Little-Endian code.
Size Notation	The following cascade menus are displayed to specify the size notation.
1 Bytes	Displays the value of the selected item as 8-bit data.
2 Bytes	Displays the value of the selected item as 16-bit data.
4 Bytes	Displays the value of the selected item as 32-bit data.
8 Bytes	Displays the value of the selected item as 64-bit data.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the address which the selected watch-expression is defined (see "(g) Jump to the address with memory definition").
Reset Color	Resets highlighting of the selected watch-expression whose value has been changed by executing a program. This item is disabled during execution of a program.
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 contents of watch-expressions").

Call Stack panel

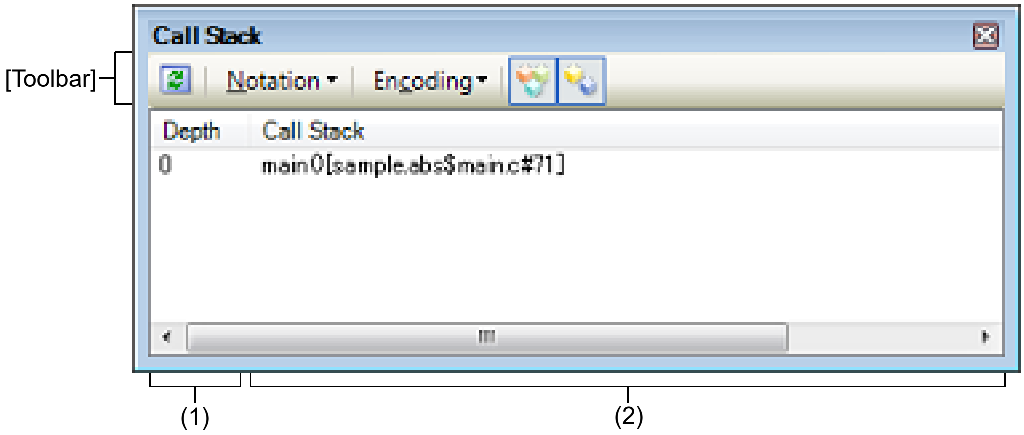
This panel is used to display the call stack information for the function call (see "2.13.1 Display call stack information"). This panel appears only when connected to the debug tool.

Caution 1. Nothing is displayed on this panel during execution of a program.
When the execution of a program is stopped, items in each area are displayed.

Caution 2. When the selected microcontroller supports multi-core, this panel displays the value regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

Remark This panel can be zoomed in and out by 100% in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Figure A.15 Call Stack Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Call Stack panel-dedicated items)]
- [[Edit] menu (Call Stack panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Call Stack].

[Description of each area]

(1) [Depth] area
The depth of the call is displayed.
The line at the current PC position becomes 0 and incremented numbers from 1 is added to the calling function in the order.

(2) [Call Stack] area
The current source position and the call stack information pushed on the stack (position of the calling function and arguments of a each function, etc.) are displayed.
The display format in this area differs depending on the selection condition of the / button on the toolbar, or of [Show Parameter]/[Show Module File Name] from the context menu.

Condition	Display Format
- Display arguments - Display module file name	*<Function>*(*<Argument>*=*<Argument Value*[Note]*>*, ...)[*<Module file name>*$*<File name>*#*<Line number>*] (default)
- Display arguments - Do not display module file name	*<Function>*(*<Argument>*=*<Argument value*[Note]*>*, ...)[*<File Name>*#*<Line number>*]
- Do not display arguments - Display module file name	*<Function>*()[*<Module file name>*$*<File name>*#*<Line number>*]
- Do not display arguments - Do not display module file name	*<Function>*()[*<File name>*#*<Line number>*]

Note When the argument value is character string, up to 20 characters can be displayed.

Remark Array arguments are passed as pointers rather than arrays (C language specification). For this reason, if the argument is an array, it is displayed as a pointer.

This area is provided with the following functions.

(a) Jump to source line and disassemble
By selecting [Jump to Source] from the context menu, the Editor panel is opened with moving the caret to the source line corresponding to the calling function at the current caret position (if the Editor panel is already open, the screen will jump to the panel).
In addition, similarly by selecting [Jump to Disassemble], the Disassemble panel (Disasemmble1) is opened with moving the caret to the address corresponding to the calling function at the current caret position (if the Disassemble panel is already open, the screen will jump to the panel (Disassemble1)).

Remark It is possible to jump to the target source line by double-clicking on that line as well.

(b) Saving the contents of call stack information
By selecting the [File] menu >> [Save Call Stack Data As...], the Save As dialog box can be opened, and all the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.13.1.4 Save the contents of call stack information" for details on the method for saving the contents of call stack information.

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel.
Notation	The following buttons to specify the notation of values are displayed.
AutoSelect	Displays values on this panel in the default notation according to the type of variable (default).
Hexadecimal	Displays values on this panel in hexadecimal number.
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.

Encoding	The following buttons to specify the encoding of character variables are displayed.
ASCII	Displays character variables in ASCII code (default).
Shift_JIS	Displays character variables in Shift-JIS code.
EUC-JP	Displays character variables in EUC-JP code.
UTF-8	Displays character variables in UTF-8 code.
UTF-16 Big-Endian	Displays character variables in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character variables in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character variables in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character variables in UTF-32 Little-Endian code.
	Displays the call stack information with the module file name (default).
	Displays the call stack information with the parameters (arguments) of the function call (default).

[[File] menu (Call Stack panel-dedicated items)]

The following items are exclusive for the [File] menu in the Call Stack panel (other items are common to all the panels). Note that all these items are disabled during execution of a program.

Save Call Stack Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of call stack information"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Call Stack Data As...].
Save Call Stack Data As...	Opens the Save As dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(b) Saving the contents of call stack information").

[[Edit] menu (Call Stack panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Call Stack panel (all other items are disabled).

Copy	Copies the contents of the selected line to the clipboard.
Select All	Selects all the items of this panel.
Find...	Opens the Find and Replace dialog box with selecting the [Find in Files] tab.
Replace...	Opens the Find and Replace dialog box with selecting the [Replace in Files] tab.

[Context menu]

Copy	Copies the contents of the selected line to the clipboard.
Show Module File Name	Displays the call stack information with the module file name (default).
Show Parameter	Displays the call stack information with the parameters (arguments) of the function call (default).

Notation	The following cascade menus to specify the notation of values are displayed.
AutoSelect	Displays values on this panel in the default notation according to the type of variable (default).
Hexadecimal	Displays values on this panel in hexadecimal number.
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.
Encoding	The following cascade menus to specify the encoding of character variables are displayed.
ASCII	Displays character variables in ASCII code (default).
Shift_JIS	Displays character variables in Shift-JIS code.
EUC-JP	Displays character variables in EUC-JP code.
UTF-8	Displays character variables in UTF-8 code.
UTF-16 Big-Endian	Displays character variables in UTF-16 Big-Endian code.
UTF-16 Little-Endian	Displays character variables in UTF-16 Little-Endian code.
UTF-32 Big-Endian	Displays character variables in UTF-32 Big-Endian code.
UTF-32 Little-Endian	Displays character variables in UTF-32 Little-Endian code.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1) and jumps to the address corresponding to the calling function of the selected line in this panel.
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the calling function of the selected line in this panel.
Jump to Local Variable at This Time	Opens the Local Variables panel to display the local variable corresponding to the selected line.

Trace panel

This panel is used to display trace data recording the execution history of the program (see "2.14 Collect Execution History of Programs").

The trace data displays by mixing the disassembled text and source text by default, but it is also possible to display either one of these by selecting the Display mode.

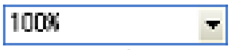
After the execution of the program is stopped, the display position is automatically updated such that the latest trace data is displayed.

This panel appears only when connected to the debug tool.

Caution 1. Software trace data can be acquired by using the Python console. See debugger.SoftwareTrace functions in "CS+ Integrated Development Environment User's Manual: Python Console" for detail.

Caution 2. **[Full-spec emulator][E1][E20]**
When trace data has been collected after selecting [all core] in the [Trace target setting] property in the [trace] category on the [Debug Tool Settings] tab of the Property panel, this panel displays the trace data regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

Remark 1. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Remark 2. This panel can be zoomed in and out by 

in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Figure A.16 Trace Panel [Full-spec emulator][E1][E20]

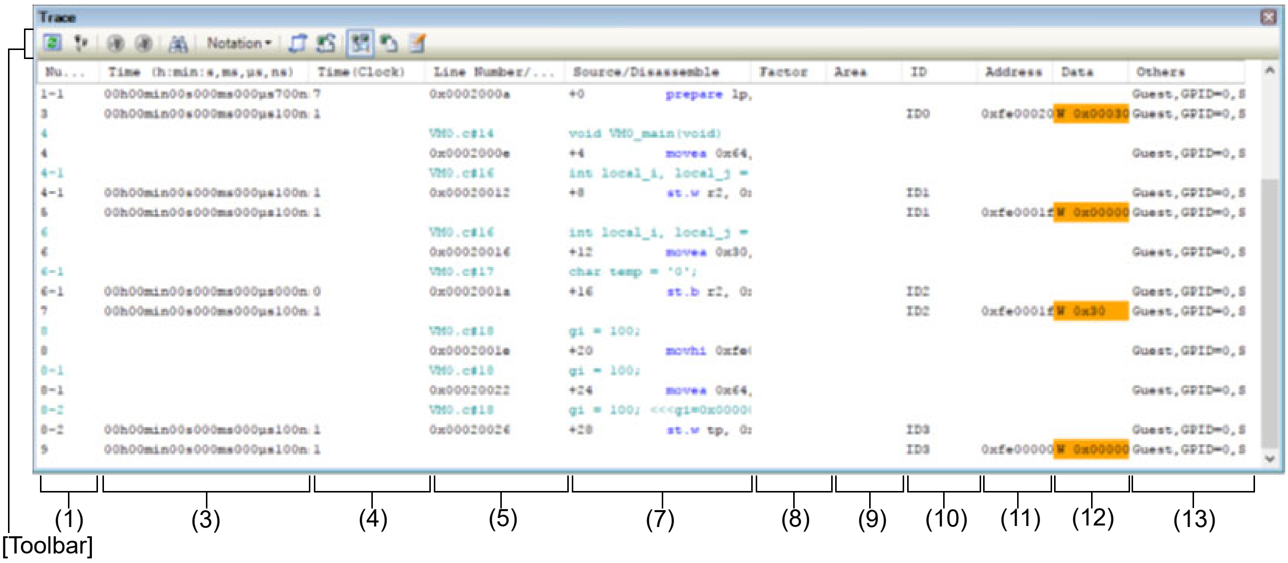
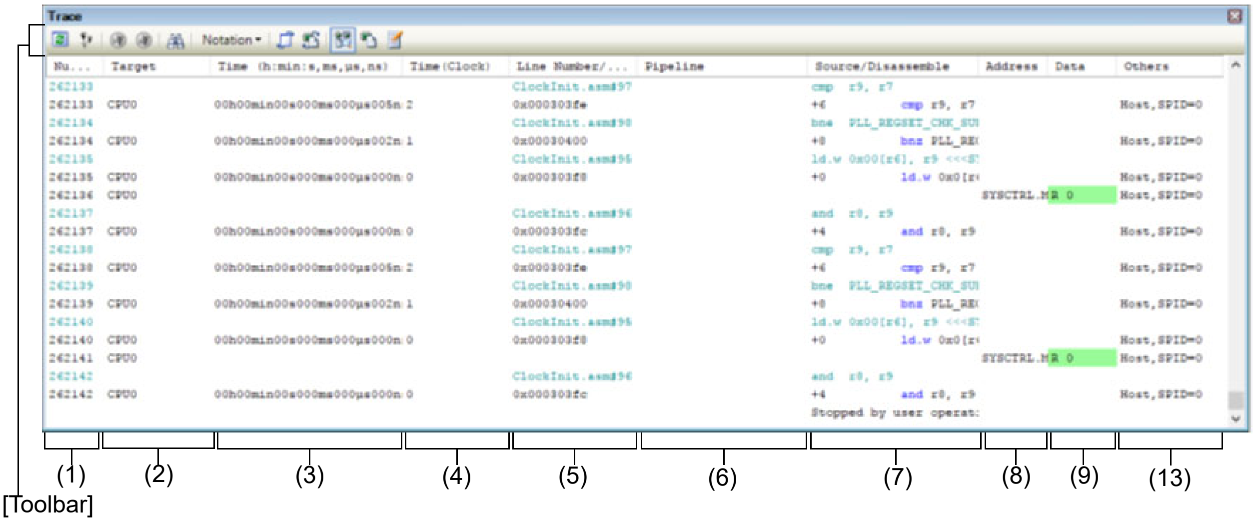


Figure A.17 Trace Panel [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [[File] menu (Trace panel-dedicated items)]
- [[Edit] menu (Trace panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Trace].
- On the Editor panel/Disassemble panel, select [Trace Settings] >> [Show Trace Result] from the context menu.

[Description of each area]

(1) [Number] area
The trace number corresponding to the trace frame is displayed.

(2) [Target] area [Simulator]
The name of the target core is displayed.

(3) [Time (h:min:s,ms,μs,ns)] area
This area displays the time required from the execution start of the program to the execution start of an instruction of each frame or generation of memory access cause.
The time is displayed in units of "hours, minutes, seconds, milliseconds, microseconds and nanoseconds".

Remark 1. **[Full-spec emulator][E1][E20]**
The time is displayed as a differential time.
When the microcontroller is multi-core, the differential time from the time of the previous data that has the same PE number is displayed.

Remark 2. **[Simulator]**
The question of whether to set the time display as an accumulated time or differential time depends on the setting of the [Accumulate trace time] property on the [Trace] category on the [Debug Tool Settings] tab of the Property panel.
When the microcontroller is multi-core and the differential time is displayed, the difference from the time of the previous data that has the same PE number is displayed.

(4) [Time(Clock)] area
This area displays the time required from the execution start of the program to the execution start of an instruction of each frame or generation of memory access cause. The time is displayed in CPU clock cycles.

Remark 1. **[Full-spec emulator][E1][E20]**
The time is displayed as a differential CPU clock cycles.

Remark 2. **[Simulator]**
The question of whether to set the time display as an accumulated CPU clock cycles or differential CPU clock cycles depends on the setting of the [Accumulate trace time] property on the [Trace] category on the [Debug Tool Settings] tab of the Property panel.

(5) [Line Number/Address] area
The line number of a source file or the address of the assemble code is displayed.
The notation of a data value can be selected by the button on the toolbar or the context menu item.
The display formats are as follows:

Type of Display Line	Display Format
Source text	*<File name>#<Line number>*
Instruction (disassemble results)	*<Address>*
Other than above	-

Remark Since the following execution histories are not displayed, the line numbers displayed are not consecutive numbers.

- CPU register access
- Operand access
- Invalid fetch

(6) [Pipeline] area [Simulator]
This area displays the pipeline execution status.
A 20-character string is displayed in this field, and each character represents the stage of the pipeline in one clock cycle. Residues of 20 from the number of clock cycles are used as indices in display of the string representing the corresponding stages of execution.
The meanings of the letters used to represent the stages are as follows.

Stage	Character
Fetch	F
Decode	D
Execute	E

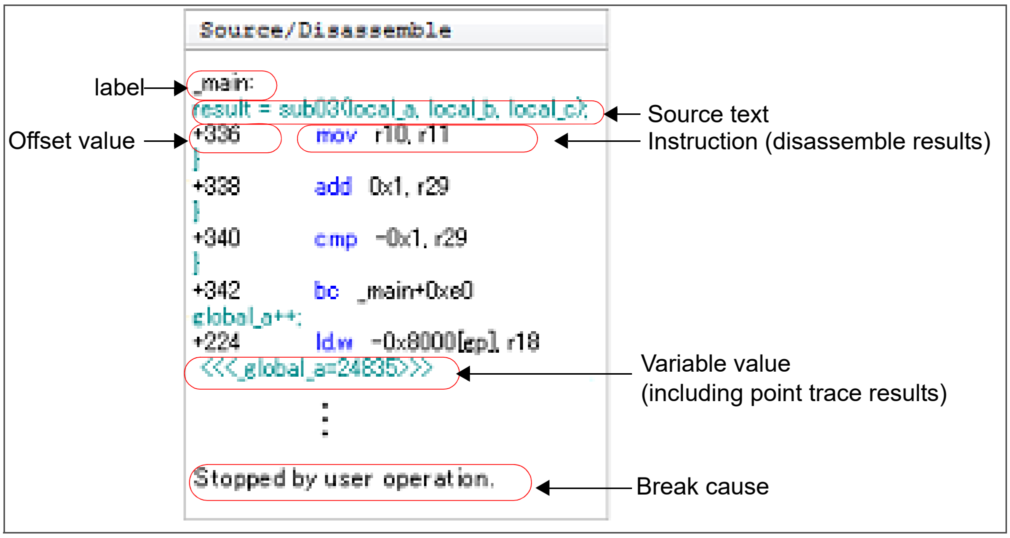
Example 1. F: 10th clock cycle, D: 11th clock cycle, E: 13th clock cycle
Display: FD_E________________

Example 2. F: 18th clock cycle, D: 19th clock cycle, E: 20th clock cycle
Display: E_________________FD

Remark "_" indicates a space.

(7) [Source/Disassemble] area
The collected trace data is displayed as follows:
Note that the items displayed in this area differ depending on the selection of the display mode (see "(a) Display mode").

Figure A.18 Display Contents of [Source/Disassemble] Area (Default)



Label	The label is displayed when a label is defined for the address.
Offset value	The offset value from the nearest label is displayed when a label is defined for the address.
Source text	The corresponding source text is displayed when the Mixed display mode or Source display mode is selected. However, when a place where no debugging information is present is executed, "<No Debug Information>" is displayed. In addition, when the value of a variable[Note 1] or an I/O register that is accessed during execution of a source line can be analyzed, that value is displayed in the following format at the end of the source line. - *<<<Variable name = Variable value>>>* - *<<<I/O register name = I/O register value>>>* Example: a=b; <<<a=5>>> The results of the Point Trace are displayed as same as format above.
Instruction (disassemble results)	The corresponding instructions are displayed as the result of disassembling when the Mixed display mode or Disassemble display mode is selected[Note 2]. The mnemonics are shown highlighted.
Break cause [Simulator]	The reason why the program has broken down is displayed.

Note 1. When there is a memory access, a symbol will be interpreted as a variable and displayed only if a symbol is assigned to the accessed address. Note, however, that only variables of up to 4 bytes are supported. If multiplication or other code is processed by the standard libraries, the label of the SADDR area used by the standard library may be shown.

Note 2. At a frame for which not all the trace data was fetched, "(LOST)" is displayed. In this case, the corresponding line is shown in error color (the error color depends on the configuration in the [General - Font and Color] category of the Option dialog box).

This area is provided with the following functions.

(a) Display mode
It is possible to select the following three display modes by selection of a button on the toolbar or the context menu.

Display Mode	Displayed Content
Mixed display mode	Displays the instruction (disassemble results), labels, source text (corresponding source line), point trace results, and break causes (default).

Display Mode	Displayed Content
Disassemble display mode	Displays the instruction (disassemble results), labels, point trace results, and break causes.
Source display mode	Displays the source text (corresponding source line) and break causes. However, when a place where no debugging information is present is executed, "<No Debug Information>" is displayed.

(b) Jumping to source line or disassemble
By selecting [Jump to Source] from the context menu, the Editor panel opens with moving the caret to the source line corresponding to the line at the current caret position (if the Editor panel is already open, the screen will jump to the panel).
In addition, similarly by selecting [Jump to Disassemble], the Disassemble panel (Disasemmble1) is opened with moving the caret to the address corresponding to the fetch address of the line at the current caret position (if the Disassemble panel is already open, the screen will jump to the panel (Disassemble1)).

(c) Linking with other panels
By clicking the / button on the toolbar, or selecting [Window Connecting] >> [Connect Source Window]/[Connect Disassemble Window] from the context menu, it is possible to link and display the corresponding places on the Editor panel/Disassemble panel, with the address of the caret position on this panel used as the pointer (no movement of the focus is done).

(d) Pop-up display
By hovering the mouse cursor over a line, all the area (item) data corresponding to that line is pop-up displayed in tandem shape.

(e) Saving trace data
The Data Save dialog box can be opened by selecting the [File] menu >> [Save Trace Data As...], and the contents of this panel can be saved in a text file (*.txt) or CSV file (*.csv).
See "2.14.9 Save the contents of execution history" for details on the method for saving trace data.

(8) [Factor] area [Full-spec emulator][E1][E20]
This area displays information on the memory access cause.
The display formats are as follows:

Factor	Display Format
PE/Thread	*<Debug target information>*
DMA	DMA
Bus Master	Bus Master ID*<Number>*[Note]
Other than above	(Nothing is displayed)

Note See User's Manual: Hardware of each device for the number.

(9) [Area] area [Full-spec emulator][E1][E20]
This area displays information on the target area of memory access.
The display formats are as follows:

Target Area	Display Format
Global RAM	Global RAM
Local RAM	Local RAM*<Number>*
Cluster RAM	Cluster RAM*<Number>*
Other than above	(Nothing is displayed)

(10) [ID] area [Full-spec emulator][E1][E20]
This area displays the ID of memory access.
The display formats are as follows:

ID	Display Format
ID number	ID *<Decimal number between 0 and 15>*
Other than above	(Nothing is displayed)

(11) [Address] area
The target address of memory access is displayed.
However, in the event of access to I/O register, the I/O register name is displayed instead of the address (when a plurality is accessed these are displayed in the following lines).
The radix of a data value can be selected by the button on the toolbar or the context menu item.

(12) [Data] area
The accessed data value and the access type at that time are displayed.
However, CPU register access is not displayed.
The notation of a data value can be selected by the button on the toolbar or the context menu item.
The display format of the data value and the access type are as follows (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box):

Display Example (Default)			Memory Access Type
R*Data value*	Character color	Standard color	Read access
	Background color	Palegreen	
W*Data value*	Character color	Standard color	Write access
	Background color	Orange	
RW*Data value*	Character color	Standard color	Read and write access
	Background color	Paleturquoise	
VECT*Data value*	Character color	Standard color	Vector read access
	Background color	Palegreen	

(13) [Other] area
Other sets of information on the virtualization facility, software tracing, and so on are displayed in this area.
Sets of information are delimited by ‘,’.
Information is given in the format <item name> or <item name> = <value>.

Type	Display Format	Display Example
Host mode	Host	Guest,GPID=0,SPID=2 Host,SPID=0
Guest mode	Guest	
Partition identifier for guest mode	GPID=<*ID*>	
System protection ID	SPID=<*ID*>	
dbcp	dbcp=<*value*>	dbcp=0x00100E4
dbpush	dbpush=(<N*umber*>:<*Value*>)	Example of the display for __dbpush(4,6) dbpush=(4:0x00000000) dbpush=(5:0x00000007) dbpush=(6:0x0001000C)
dbtag	dbtag=<*Value*>	dbtag=0x3FF

[Toolbar]

	Acquires the latest data from the debug tool, and updates the contents of this panel. This item is disabled while the tracer is running.
	Clears the trace memory and the display of this panel (initialized). This item is disabled while the tracer is running.
	Starts the tracer operation. The content currently being displayed in this panel is cleared. This item is disabled while the tracer is running.
	Stops the tracer operation. The contents of trace data newly acquired are displayed. This item is disabled while the tracer is stopped.
	Opens the Trace Search dialog box.
Notation	The following buttons to change the notation of a data value are displayed. This item is disabled while the tracer is running.
	Displays values on this panel in hexadecimal number (default).
	Displays values on this panel in decimal number.
	Displays values on this panel in octal number.
	Displays values on this panel in binary number.
	Links with the Editor panel.
	Links with the Disassemble panel.
	Sets to the Mixed display mode as the display mode (default). This item is disabled while the tracer is running.
	Sets to the Disassemble display mode as the display mode. This item is disabled while the tracer is running.
	Sets to the Source display mode as the display mode. This item is disabled while the tracer is running.

[[File] menu (Trace panel-dedicated items)]

The following items are exclusive for the [File] menu in the Trace panel (other items are common to all the panels).
Note that all these items are disabled during execution of a program.

Save Trace Data	Overwrites the contents of this panel to the previously saved text file (*.txt)/CSV file (*.csv) (see "(e) Saving trace data"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Trace Data As...]. This item is disabled while the tracer is running.
Save Trace Data As...	Opens the Data Save dialog box to newly save the contents of this panel to the specified text file (*.txt)/CSV file (*.csv) (see "(e) Saving trace data"). This item is disabled while the tracer is running.

[[Edit] menu (Trace panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Trace panel (all other items are disabled).

Copy	Copies the contents of the selected line to the clipboard (multiple line selections impossible). This item is disabled while the tracer is running.
Find...	Opens the Trace Search dialog box.

[Context menu]

Clear Trace	Clears the trace memory and the display of this panel (initialized). This item is disabled while the tracer is running.
Start Trace	Starts the tracer operation (see "2.14.5.2 Restart collection of execution history"). The content currently being displayed in this panel is cleared. This item is disabled while the tracer is running.
Stop Trace	Stops the tracer operation (see "2.14.5.1 Stop collection of execution history temporarily"). The contents of trace data newly acquired are displayed. This item is disabled while the tracer is stopped.
Find...	Opens the Trace Search dialog box. This item is disabled while the tracer is running.
Copy	Copies the contents of the selected line to the clipboard (multiple line selections impossible). This item is disabled while the tracer is running.
Mixed Display	Sets to the Mixed display mode as the display mode. This item is disabled while the tracer is running.
Disassemble View	Sets to the Disassemble display mode as the display mode. This item is disabled while the tracer is running.
Source View	Sets to the Source display mode as the display mode. This item is disabled while the tracer is running.
Notation	The following cascade menus are displayed to specify the notation. This item is disabled while the tracer is running.
Hexadecimal number	Displays values on this panel in hexadecimal number (default).
Decimal	Displays values on this panel in decimal number.
Octal	Displays values on this panel in octal number.
Binary	Displays values on this panel in binary number.
Window Connecting	The following cascade menus are displayed to link with other panels (see "(c) Linking with other panels").
Connect Source Window	Links with the Editor panel.
Connect Disassemble Window	Links with the Disassemble panel.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1) and jumps to the fetch address corresponding to the line at the caret position in this panel.
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the line at the caret position in this panel.
Jump to Memory	Opens the Memory panel and jumps to the memory value corresponding to the line at the caret position in this panel.

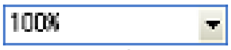
Events panel

This panel is used to display the detailed information about the events that are set on the Editor panel/Disassemble panel/Watch panel. On this panel, you can change the setting state of the event between valid/invalid and delete the event (see "2.19 Manage Events").

This panel appears only when connected to the debug tool.

Remark 1. Also see "2.19.7 Notes for setting events" for details on events (e.g. limits on the number of enabled events).

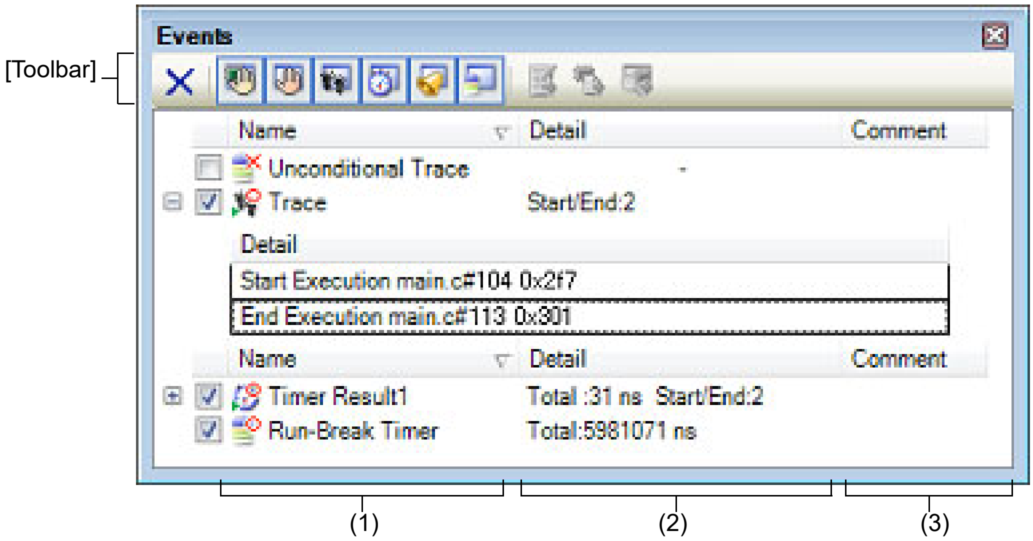
Remark 2. Events set via the Function List panel or Variable List panel of the analyze tool (Program Analyzer) are also managed on this panel.

Remark 3. This panel can be zoomed in and out by 

in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Remark 4. When the separator line of each area in this panel is double-clicked, the width of the area changes to the shortest possible size that can display the contents of the area.

Figure A.19 Events Panel



This section describes the following.

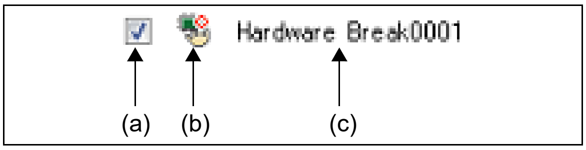
- [How to open]
- [Description of each area]
- [Toolbar]
- [[Edit] menu (Events panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Event].
- **[Simulator]**
  On the Editor panel/Disassemble panel, select [Timer Settings] >> [View Result of Timer] from the context menu.

[Description of each area]

(1) [Name] area
A list of the event names that have currently been set is displayed in the form shown below.



Remark It is possible to limit the event to be displayed by clicking the button on the toolbar (see "[Toolbar]").

(a) Check box
The setting state of the event is displayed/changed.
Note that the Event mark is changed depending on the setting state of the event.

	Valid state	Event occurs when the specified condition is met. It is possible to set the event to an invalid state by removing the check.
	Invalid state	Event does not occur when the specified condition is met. It is possible to set the event to a valid state by removing the check.
	Suspended state	The conditions that have been specified cannot be set with the program of the debugging target. It is not possible to operate the check box.

Remark 1. Both of the Timer Start event and Timer Stop event is must be set for the Timer Result event. Therefore, it is not possible to set a particular event to a valid state by only the setting of one of these (at the same time as both events are set, they are treated as grouped events as a Timer Result).

Remark 2. It is not possible to set the Run-Break Timer event to an invalid/suspended state.

Remark 3. The setting of the Unconditional Trace event and the Trace event to valid or invalid state is exclusively controlled. Therefore, the Unconditional Trace event, which is a built-in event, is valid state by default, but if either a trace start event/trace end event is set, it automatically becomes invalid state, and the Trace event, which is a event name that is collectively called with a trace start event and a trace end event, becomes valid state. Conversely, if the set Trace event is invalid state, the Unconditional Trace event automatically becomes valid state.

(b) Event mark
The event mark shows the type of event, and in addition shows the current setting state.
The meanings of the marks displayed are as follows:

Table A.8 Event Mark

Event Type	Valid State	Invalid State	Suspended State	Note
Hardware Break				Including a hardware break point
Software Break				Including a software break point
Break at start of function				A break event that can be set via the analyze tool.
Access break to variable				
Unconditional Trace			None	-
Run-Break Timer		None	None	-
Trace				Displayed on only the Events panel
Trace start				Displayed on only the Editor panel/ Disassemble panel
Trace end				
Timer Result				Displayed on only the Events panel
Timer start				Displayed on only the Editor panel/ Disassemble panel
Timer end				

Event Type	Valid State	Invalid State	Suspended State	Note
Performance Measurement				Displayed on only the Events panel
Performance measurement start				Displayed on only the Editor panel/ Disassemble panel
Performance measurement end				
Point Trace				-
Printf (Action event)				-
Setting of two or more events	Note 1	Note 2	Note 3	Displayed on only the Editor panel/ Disassemble panel

Note 1. There is one or more event with valid state.

Note 2. There is no event with valid state and at least one event with invalid state.

Note 3. All the set events are suspended state.

(c) Event name
The event type and ID number are displayed as the event name.
A number from 0001 is automatically provided as the ID number for each event (no renumbering of the ID number is done even in the event that an event that has been set once is deleted).
Event types that are displayed are as follows:

Table A.9 Event Type

Event Type	Description
Hardware Break (Break[Note 1])	Breaks the program when the condition is met while the debug tool monitors the break condition all the time during program execution. -> See "2.11.3 Stop the program at the arbitrary position (breakpoint)" -> See "2.11.4 Stop the program at the arbitrary position (break event)" -> See "2.11.5 Stop the program with the access to variables/I/O registers"
Software Break (Break[Note 1])	Breaks the program when the instruction, which an address code to break is rewritten for the break instruction, is executed. -> See "2.11.3 Stop the program at the arbitrary position (breakpoint)"
Break at start of function	This event type is a Hardware Break (execution type) that is set in the Function panel of the analyze tool (Program Analyzer).
Access break to variable	This event type is a Hardware Break (access type) that is set in the Variable panel of the analyze tool (Program Analyzer).
Unconditional Trace	Automatically collects the trace data with start of a program execution, and stops collecting the trace data with stop of the program execution. This event cannot be deleted because of the built-in event[Note 2] (this event is set to a Valid state by default). -> See "2.14.2 Collect execution history until stop of the execution"
Run-Break Timer	Automatically measures the execution time of a program with start of the program execution, and stops the measurement with stop of the program execution. This event cannot be deleted because of the built-in event[Note 2] (this event is set to a Valid state by default). -> See "2.15.1 Measure execution time until stop of the execution"
Trace	Starts/stops collecting the trace data when the condition specified with a trace start event and a trace end event is met (this event is displayed when either a trace start event or a trace end event is set). -> See "2.14.3 Collect execution history in a section"

Event Type	Description
Timer Result *n*	Starts/stops measuring the execution time of a program when the condition specified with a timer start event and a timer end event is met (this event is displayed when either a timer start event or a timer end event is set). "*n*" indicates the channel number in which a Timer Result event is set. -> See "2.15.2 Measure execution time in a section"
Performance Measurement *n*	Starts/stops performance measurement when the condition specified with a performance measurement start event and a performance measurement end event is met (this event is displayed when either a performance measurement start event or a performance measurement end event is set). "*n*" indicates the channel number in which a Performance Measurement event is set. -> See "2.16.1 Measure the performance in a section"
Point Trace	Records the information as the trace data only when accessing the specified variable or I/O register during execution of a program. -> See "2.14.4 Collect execution history only when the condition is met"
Printf	Executes printf command in software processing after temporary stopping a program in execution at an arbitrary position (action event). -> See "2.18.1 Inset printf"

Note 1. A breakpoint that is set by a one click operation of the mouse is displayed "Break" (see "2.11.3.1 Set a breakpoint").

Note 2. This is set in the debug tool by default.

(2) [Detail Information] area
Detailed information about each event is displayed.
The contents of the information that is displayed differ depending on the event type as follows:

Table A.10 Detailed Information with Event Type

Event Type	Displayed Content[Note 1]					
Hardware Break (Condition: execution)	Format1	*<PE> <Condition to occur> <File name#Line number> <Address>*				
	Example	CPU1	Before Execution	main.c#39	0x100	
		CPU1	After Execution	sub.c#100	0x200	
		CPU1	Before Execution	-	0x300	
		CPU1	Execution	main.c#39	0x300	**[Simulator]**
	Format2	*<PE> <Condition to occur> <Symbol + Offset> <Address>*				
	Example	CPU1	Before Execution	funcA + 0x10	0x100	
		CPU1	After Execution	funcB + 0x20	0x200	
		CPU1	Before Execution	-	0x300	

Event Type	Displayed Content[Note 1]	
Hardware Break (Condition: access)	Format1	*<PE> <Condition to occur> <File name#Variable name> <Address(range)> <Comparison condition> <Comparison value>*
	Example	CPU1 Read main.c#variable1 0x100 - 0x101 == 0x5
		CPU1 Write sub.c#variable2 0x200 - 0x200 == 0x7
		CPU1 Read/Write sub2.c#variable3 0x300 - 0x303 == 0x8
	Format2	*<PE> <Condition to occur> <File name#Function name#Variable name> <Address(range)> <Comparison condition> <Comparison value>*
	Example	CPU1 Read main.c#func1#variable1 0x100 - 0x101 == 0x10
	Format3	*<PE> <Condition to occur> <Variable name> <Address(range)> <Comparison condition> <Comparison value>*
	Example	CPU1 Write variable1 0x100 - 0x101 == 0x10
Software Break	Format1	*<Condition to occur> <File name#Line number> <Address>*
	Example	Before Execution main.c#40 0x102
		Before Execution sub.c#101 0x204
	Format2	*<Condition to occur> <Symbol + Offset> <Address>*
	Example	Before Execution funcA + 0x12 0x102
Unconditional Trace	Format	-
	Example	-
Run-Break Timer	Format	Total: *<Total execution time>*
	Example	Total: 1000ms
		Total: OVERFLOW
Trace (Condition: execution)	Format	Total of Start/End: *<Total number of trace start/trace end events>* [Note 2] *<PE>* <Start/End> *<Detailed information of trace start/trace end event>*
	Example	Total of Start/End: 4 - CPU1 Start After Execution main.c#100 0x300 - CPU1 Start After Execution funcA + 0x100 0x300 - CPU1 End After Execution main.c#200 0x100 - CPU1 End After Execution funcA + 0x10 0x100
Timer Result *n* (Condition: execution)	Format	Total:*<Total execution time >* Total of Start/End: *<Total number of timer start event/timer end event>*[Note 2] - *<Total execution time> <Pass Count> <Average> <Max> <Min>* - *<PE> <Start/End> <Detailed information of timer start event/timer end event>*
	Example	Total: 10ms Total of Start/End: 4 - Total: 10ms Pass Count: 5 Average: 2ms Max: 4ms Min: 1ms - CPU1 Start After Execution main.c#100 0x300 - CPU1 Start After Execution funcA + 0x30 0x100 - CPU1 End After Execution main.c#100 0x300 - CPU1 End After Execution funcA + 0x50 0x100

Event Type	Displayed Content[Note 1]	
Performance Measurement *n* (Condition: execution)	Format	*<Performance measurement mode> <Performance measurement result> <Total number of performance measurement start event/performance measurement end event>* - *<PE> <Start/End> <Detailed information of performance measurement start event/performance measurement end event>*
	Example	ALL instruction count Count:10 Total of Start/End: 2 - CPU1 Start After Execution main.c#100 0x300 - CPU1 End After Execution main.c#100 0x300
Performance Measurement *n* (Condition: access)	Format1	*<Performance measurement mode> <Performance measurement result> <Total number of performance measurement start event/performance measurement end event>* - *<PE> <Start/End> <Condition to occur> <File name#Function name#Variable name> <Address(variable range)> <Comparison condition symbol> <Comparison value>*
	Example	ALL instruction count Count:10 Total of Start/End: 2 - CPU1 Start Read main.c#variable1 0x100 – 0x101 == 0x5 - CPU1 End Write sub.c#variable2 0x200 – 0x200 == 0x7
	Format2	*<Performance measurement mode> <Performance measurement result> <Total number of performance measurement start event/performance measurement end event>* - *<PE> <Start/End> <Condition to occur> <File name#Function name#Variable name> <Address(variable range)> <Comparison condition symbol> <Comparison value>*
	Example	ALL instruction count Count:10 Total of Start/End: 2 - CPU1 Start Read main.c#func1#variable1 0x100 – 0x101 == 0x10 - CPU1 End Write main.c#func1#variable1 0x100 – 0x101 == 0x10
	Format3	*<Performance measurement mode> <Performance measurement result> <Total number of performance measurement start event/performance measurement end event>* - *<PE> <Start/End> <Condition to occur> <Variable name> <Address(variable range)> <Comparison condition symbol> <Comparison value>*
	Example	ALL instruction count Count:10 Total of Start/End: 2 - CPU1 Start Read variable1 0x100 – 0x101 == 0x10 - CPU1 End Write variable1 0x100 – 0x101 == 0x10
Point Trace (Condition: access)	Format1	*<PE> <Condition to occur> <Variable name> <Variable address>*
	Example	CPU1 Read variable1 0x100
	Format2	*<PE> <Condition to occur> <File name#Variable name> <Variable address>*
	Example	CPU1 Write sub.c#variable2 0x200
	Format3	*<PE> <Condition to occur> <File name#Function name#Variable name> <Variable address>*
	Example	CPU1 Read/Write sub.c#func1#variable3 0x300

Event Type	Displayed Content[Note 1]		
Printf (Action event)	Format	*<Condition to occur> <File name#Line number> <Address> <Setting of Printf event>*	
	Example	Before Execution	main.c#39 0x100 aaa, bbb, ccc
		After Execution	sub.c#100 0x200 Result of aaa : aaa

Note 1. Following are the details on the display format.

<PE>	If the microcontroller is multi-core, the core name is shown. If the microcontroller is single-core, nothing is shown.
<Condition to occur>	Displays one of the following conditions. [Full-spec emulator][E1][E20] Execution: Before Execution or After Execution Access: Before Read, After Read, Before Write, After Write, Before Read/Write, After Read/Write [Simulator] Execution: Before Execution Access: Read, Write, Read/Write
<File name#Line number>	Shows the line number of the source. Display format is the same as the watch type scope specification expression. When multiple load module files are downloaded, *<Load module file name$File name#Line number>* is displayed. For those events set in the Disassemble panel, display *<Line number>* in the format *<Symbol + offset>* in the condition below. - Line information exists and the specified position that the event is set not the top of the line information - Line information does not exist and symbol information exists. Show *<Line number>* in "-" in the following condition. - Line information and symbol information does not exist.
<Variable name>	Shows the variable name in the source file. Display format is the same as the watch type scope specification expression.
<Comparison condition>	Condition to compare (==) is shown. If the comparison value is not specified, comparison condition is not shown.
<Comparison value>	Comparison value is shown. If the comparison value is not specified, comparison condition is not shown.
<Address>	Address in the memory area is shown (only in hex number).
<Start/End>	Shows whether the contents of the detailed information is start event or the stop event.
<Pass Count>	Shows the measurement result of the pass count of the timer. If a timer overflow occurs (see "2.15.3 Measurable time"), or if the illegal value was acquired, "OVERFLOW" is displayed. If measurements have not been performed yet, "Not measured" is displayed.
<Total>	Shows the measurement result of the timer total execution time. The unit is either of ns/µs,/ms/s/min/clock (if, however, the unit is in "min", a value in "s" unit also appears). If a timer overflow occurs (see "2.15.3 Measurable time"), or if the illegal value was acquired, "OVERFLOW" is displayed. If measurements have not been performed yet, "Not measured" is displayed.

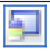
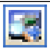
<Average>	Shows the measurement result of average execution of the timer. The unit is either of ns/µs,/ms/s/min/clock (if, however, the unit is in "min", a value in "s" unit also appears). If a timer overflow occurs (see "2.15.3 Measurable time"), or if the illegal value was acquired, "OVERFLOW" is displayed. If measurements have not been performed yet, "Not measured" is displayed.
<Max>	Shows the measurement result of the maximum execution time of the timer. The unit is either of ns/µs,/ms/s/min/clock (if, however, the unit is in "min", a value in "s" unit also appears). If a timer overflow occurs (see "2.15.3 Measurable time"), or if the illegal value was acquired, "OVERFLOW" is displayed. If measurements have not been performed yet, "Not measured" is displayed.
<Min>	Shows the measurement result of the minimum execution time of the timer. The unit is either of ns/µs,/ms/s/min/clock (if, however, the unit is in "min", a value in "s" unit also appears). If a timer overflow occurs (see "2.15.3 Measurable time"), or if the illegal value was acquired, "OVERFLOW" is displayed. If measurements have not been performed yet, "Not measured" is displayed.
<Set print event>	Shows the variable expression and the character strings specified in the Action Events dialog box.
<Performance measurement mode>	Shows the mode of performance measurement. The mode set in the Detailed Settings of Performance Measurement dialog box [Full-spec emulator][E1][E20] is displayed.
<Performance measurement result>	Shows the result of performance measurement. If "Clock cycle", "Non-interrupt cycle", or "Interrupt disable cycle of DI/EI" is set as the performance measurement mode in the Detailed Settings of Performance Measurement dialog box [Full-spec emulator][E1][E20], the number of cycles is displayed, otherwise the number of counts is displayed.

Note 2. Click this line to display the detailed information of the lower lines.

(3) [Comment] area
The user can write comments for each event that has been set.
To input comments, click on this area, or select [Edit Comment] form the context menu after selecting the event in which you want to input comments, and then input directly the desired text from the keyboard (the edit mode is cancelled by pressing down the [Esc] key).
After editing the comments, complete the editing by pressing the [Enter] key or moving the focus to outside the edit region. Up to a maximum of 256 characters can be inputted for the comments, and this is saved as the settings of the user during use.

[Toolbar]

	Deletes the selected event and event condition. Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).
	Displays events related to Hardware Break (default).
 [Full-spec emulator] [E1][E20]	Displays events related to Software Break (default).

	Displays events related to the trace (default).
	Displays events related to the timer (default).
	Displays events related to performance measurement (default).
	Displays events related to the action event (Printf event) (default).
	Displays events related to the built-in event (Unconditional Trace event/Run-Break Timer event) (default).
	Opens the Editor panel and jumps to the source line corresponding to the address where the selected event[Note] is being set.
	Opens the Disassemble panel and jumps to the disassemble results corresponding to the address where the selected event[Note] is being set.
	Opens the Memory panel and jumps to the memory corresponding to the address where the selected event[Note] is being set.

Note Events other than Trace events, Timer Result events and built-in events (Unconditional Trace events/ Run-Break Timer events) can be objects of this button.

[[Edit] menu (Events panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Events panel (all other items are disabled).

Delete	Deletes the selected event and event condition. Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).
Select All	Selects all the events displayed on the panel.
Find...	Opens the Find and Replace dialog box with selecting [Find in Files] tab.
Replace...	Opens the Find and Replace dialog box with selecting [Replace in Files] tab.

[Context menu]

Enable Event	Enables the selected event (valid state). Note that this item is disabled if the selected event is a valid state.
Disable Event	Disables the selected event (invalid state). Note that this item is disabled if the selected event is an invalid state.
Delete	Deletes the selected event. Note that it is not possible to delete the built-in events (Unconditional Trace event and Run-Break Timer event).
Select All	Selects all the events of this panel.

View Select	The following cascade menus are displayed to limit the event type to be displayed. All of the items have been selected by default.
Hardware Break	Displays events related to Hardware Break.
Software Break	Displays events related to Software Break.
Timer Event	Displays events related to the timer.
Performance Measurement Event	Displays events related to performance measurement.
Trace Event	Displays events related to the trace.
Action Event	Displays events related to action events (Printf events).
Built-in Event	Displays events related to built-in events (Unconditional Trace event or Run-Break Timer event).
Timer Settings	The following cascade menus are displayed to do the settings related to the timer. Note that this item is enabled only when a timer-related event has been selected.
Init Timer	Initializes the timer used by the selected event (except for Run-Break Timer).
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.
Clock	Displays the result of a selected event measured by a timer in clock units.
Init Performance Measurement	Initializes performance measurement used by the selected event.
Jump to Memory	Opens the Memory panel (Memory1) and jumps to the memory corresponding to the address where the selected event[Note] is being set.
Jump to Disassemble	Opens the Disassemble panel (Disassemble1) and jumps to the disassemble results corresponding to the address where the selected event[Note] is being set.
Jump to Source	Opens the Editor panel and jumps to the source line corresponding to the address where the selected event[Note] is being set.
Edit Condition...	Opens one of the following dialog box to edit the selected event - For an action event (Printf event) Action Events dialog box
Edit Comment	Sets to the edit mode to input comments for the selected event. When comments are already present, all of that character string is set to a select state.

Note Events other than Trace events, Timer Result events and built-in events (Unconditional Trace events/ Run-Break Timer events) can be objects of this item.

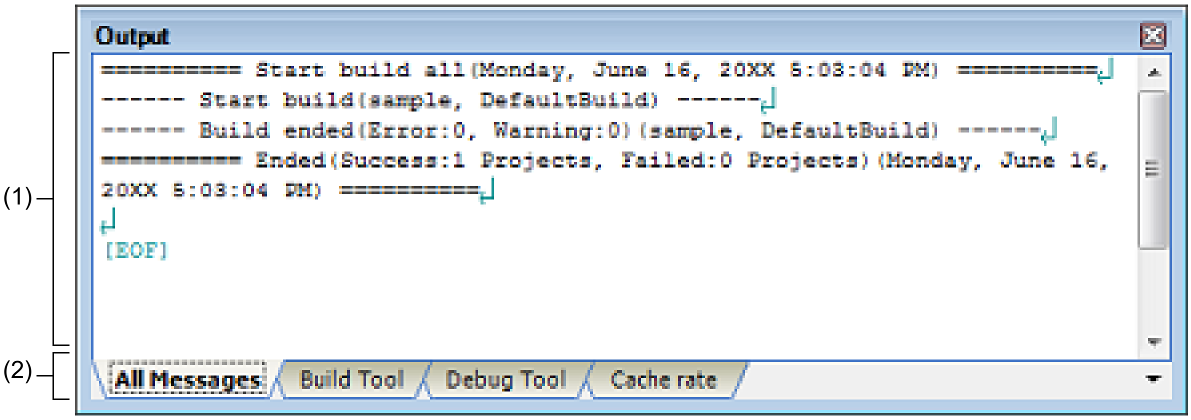
Output panel

This panel is used to display operation logs for various components (debug tool, design tool, build tool, etc.) provided by CS+, in addition to results of batch searches by the Find and Replace dialog box and a Printf event (see "2.18.1 Inset printf").

The messages are classified by the message origination tool and displayed on the individual tabs.

Remark This panel can be zoomed in and out by [100%] in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Figure A.20 Output Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [[File] menu (Output panel-dedicated items)]
- [[Edit] menu (Output panel-dedicated items)]
- [Context menu]

[How to open]

- From the [View] menu, select [Output].

[Description of each area]

(1) Message area

The output messages of each tool, search results and results by a Printf event are displayed.

In the case of search results (batch search), every time a search is performed, a new message will be displayed after the previous message is cleared (except for the [All Messages] tab).

The colors of message display differ with the type of message as shown below (character colors and background colors depend on the configuration in the [General - Font and Color] category of the Option dialog box).

Message Type	Display Example (Default)			Description
Normal message	AaBbCc	Character color	Black	Displayed with information notices
		Background color	White	
Warning message	AaBbCc	Character color	Blue	Displayed with warnings about operations
		Background color	Standard color	
Error message	AaBbCc	Character color	Red	Displayed when there is a critical error, or when execution is not possible due to a operational mistake
		Background color	Light gray	

This area is provided with the following functions.

(a) Tag jump
By double-clicking on the output message, the Editor panel is opened and the number of the corresponding line in the corresponding file is displayed.
This allows you to jump from error messages that are output when building, etc. to the corresponding error line in the source file.

(b) Help display
If there is a caret on the line where a warning message or error message is being displayed, you can select [Help for Message] from the context menu. You can also display help for that line's message by pressing the [F1] key.

(c) Saving a log
The Save As dialog box can be opened by selecting the [File] menu >> [Save Output-*tab name* As...], and the contents that are displayed on the currently selected tab can be saved in a text file (*.txt) (messages on deselected tabs will not be saved).

(2) Tab selection area
Select the tab that indicates the origin of message.
The following tabs are available for the debug tool.

Tab Name	Description
All Messages	Displays operation logs for all components (debug tool, design tool, build tool, etc.) provided by CS+ in order of output.
Debug Tool	Displays messages output from the debug tool. Display only operation logs for the debug tool out of those for various components (debug tool, design tool, build tool, etc.) provided by CS+.
Cache rate [Simulator]	Displays the cache hit rate (the ratio of the cache hit count to the cache access count).
Find and Replace	Displays the batch search results from the Find and Replace dialog box.

Caution Even if a new message is output on a deselected tab, tab selection will not automatically switch. In this case, "*" mark will be added in front of the tab name, indicating that a new message has been output.

[[File] menu (Output panel-dedicated items)]

The following items are exclusive for the [File] menu in the Output panel (other items are common to all the panels).
Note that all these items are disabled during execution of a program.

Save Output-*tab name*	Overwrites the contents that are displayed on the currently selected tab to the preciously saved text file (see "(c) Saving a log"). Note that when the file has never been saved or the file is write disabled, the same operation is applied as the selection in [Save Output-*tab name* As...]. This item is disabled while building.
Save Output-*file name* As...	Opens the Save As dialog box to newly save the contents that are displayed on the currently selected tab to the specified text file (*.txt) (see "(c) Saving a log").

[[Edit] menu (Output panel-dedicated items)]

The following items are exclusive for [Edit] menu in the Output panel (all other items are disabled).

Copy	Copies the contents of the selected range to the clipboard as character string(s).
Select All	Selects all the messages displayed on the currently selected tab.
Find...	Opens the Find and Replace dialog box with selecting [Quick Find] tab.
Replace...	Opens the Find and Replace dialog box with selecting [Replace in Files] tab.

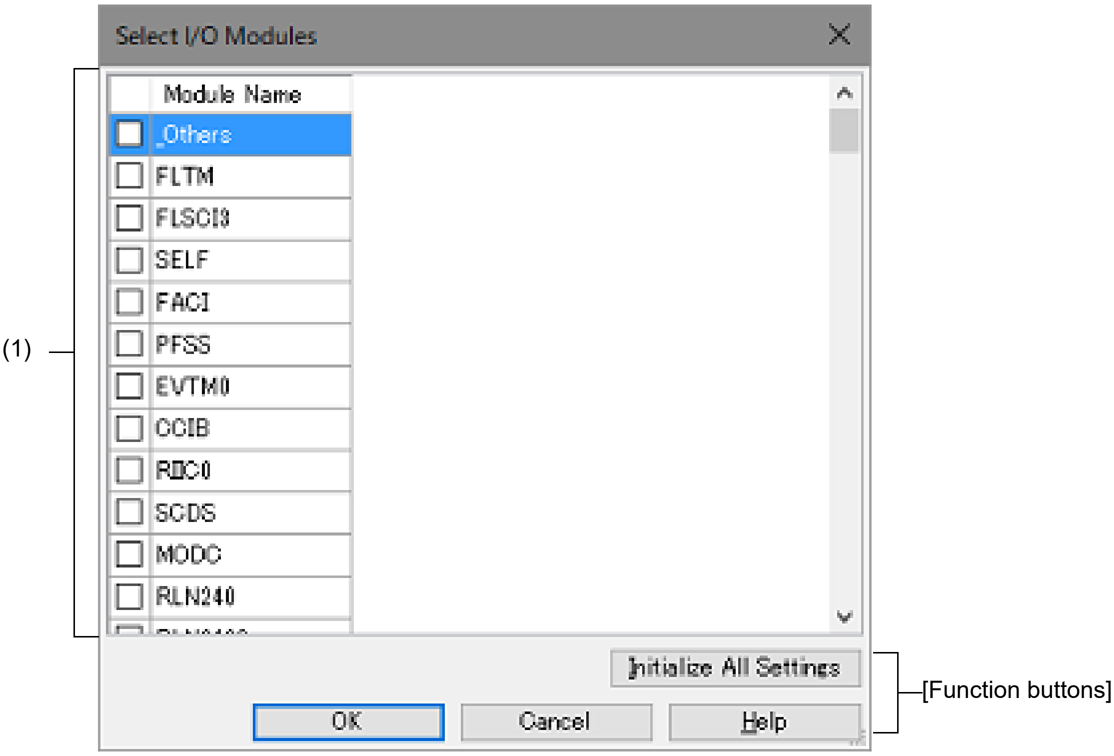
[Context menu]

Copy	Copies the contents of the selected range to the clipboard as character string(s).
Select All	Selects all the messages displayed on the currently selected tab.
Clear	Deletes all the messages displayed on the currently selected tab.
Tag Jump	Opens the Editor panel and jumps to the number of the corresponding line in the corresponding file of the message at the caret position.
Stop Searching	Cancels the search currently being executed. This item is disabled when a search is not being executed.
Help for Message	Displays help for the massage on the current caret position. This item only applies to warning messages and error messages.

Select I/O Modules dialog box

This dialog box is used to set the I/O modules which details are displayed.

Figure A.21 Select I/O Modules Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Debug Tool Settings] tab of the Property panel, click the [...] button displayed by selecting one of the values of the [I/O modules] property in the [Memory] category.

[Description of each area]

(1) Area for specifying I/O module name
This area displays a list of the names of I/O modules defined in the device file.

(a) [Module Name]
This area displays the names of I/O modules which are defined in the device file.
If any of the check boxes is selected, the I/O modules are used for debugging.

[Function buttons]

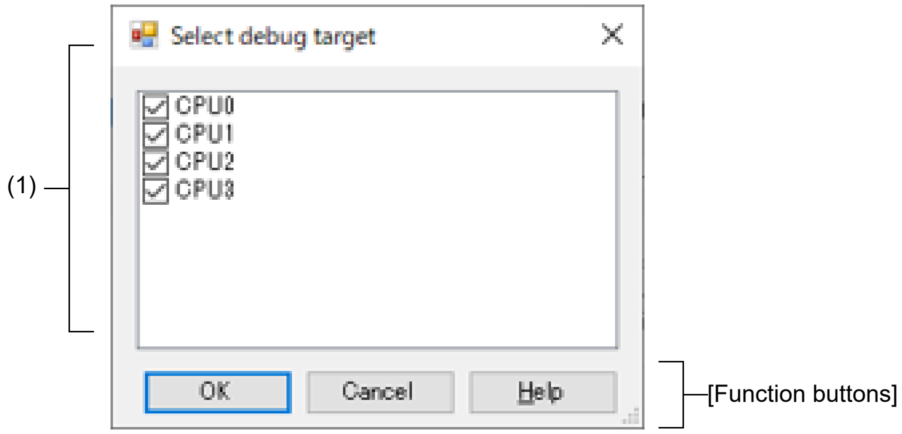
Button	Function
OK	Closes this dialog box and calls the settings to reflect them in the previous properties.

Button	Function
Initialize All Settings	Sets all check boxes under [Module Name] settings to their default values.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help of this dialog box.

Select Debug target dialog box

This dialog box is used to set the debugging.

Figure A.22 Select Debug target Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Connect Settings] tab of the Property panel, click the [...] button displayed by selecting one of the values of the [Debug target] property in the [Multi-core] category.

[Description of each area]

(1) Area with a list of contexts for debugging
This area displays a list of contexts for debugging.
When a checkbox is selected, the selected context is handled as a target for debugging.
The checkboxes are selected by default.

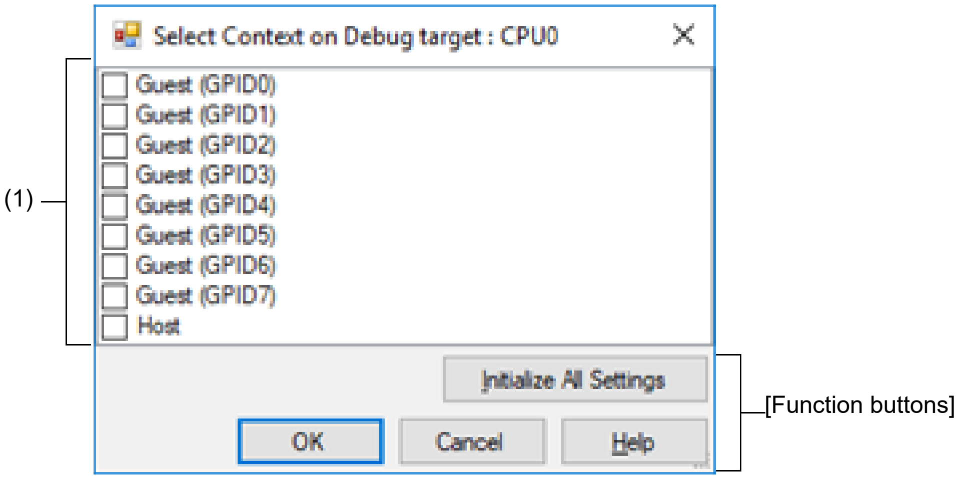
[Function buttons]

Button	Function
OK	Completes the settings of contexts for debugging and closes this dialog box.
Cancel	Cancels the modification of contexts for debugging and closes this dialog box.
Help	Displays the help of this dialog box.

Select Contexts on Debug target dialog box

This dialog box is used to set the contexts for debugging.

Figure A.23 Select Contexts on Debug target Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Debug Tool Settings] tab of the Property panel, click the [...] button displayed by selecting one of the values of the [Contexts on debug target] property in the [Hardware-assisted Virtualization] category.

[Description of each area]

(1) Area with a list of contexts for debugging
This area displays a list of contexts for debugging.
When a checkbox is selected, the selected context is handled as a target for debugging.
The checkboxes are deselected by default.

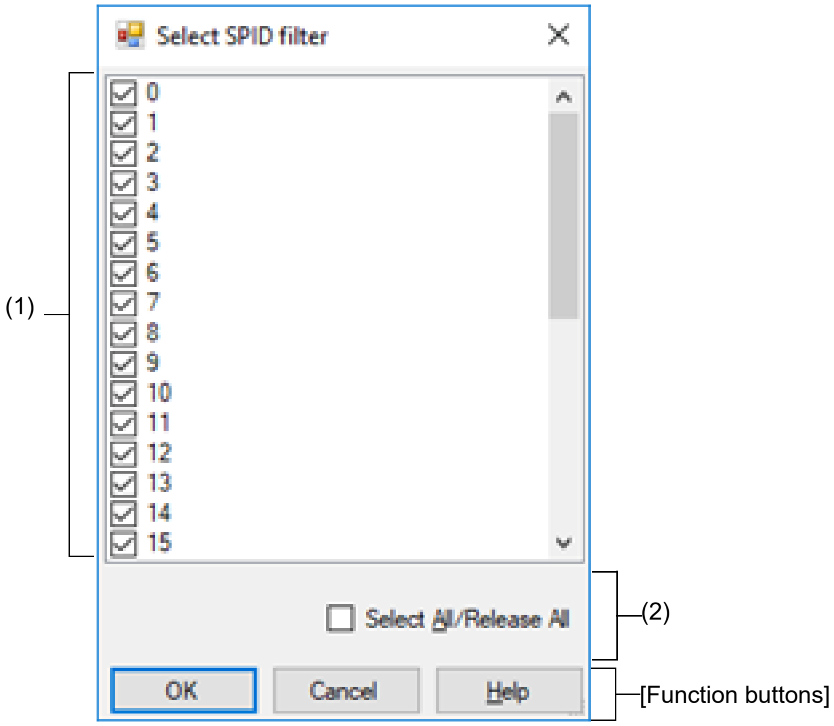
[Function buttons]

Button	Function
OK	Completes the settings of contexts for debugging and closes this dialog box.
Initialize All Settings	Sets all checkboxes of contexts for debugging to their default state (deselected).
Cancel	Cancels the modification of contexts for debugging and closes this dialog box.
Help	Displays the help of this dialog box.

Select SPID filter dialog box

This dialog box is used to set the SPID filter.

Figure A.24 Select SPID filter Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Debug Tool Settings] tab of the Property panel, click the [...] button displayed by selecting one of the values of the [SPID filter] property in the [Trace] category.

[Description of each area]

(1) Area with a list of SPIDs
This area displays a list of SPIDs.
When a checkbox is selected, trace information for the SPID with the given number is collected. The checkboxes are selected by default.

(2) Area for the [Select All/Release All] checkbox

(a) Select All/Release All checkbox
All checkboxes in the area with the list of SPIDs are selected or deselected.
Selecting this checkbox selects all checkboxes in the area with the list of SPIDs.
Deselecting this checkbox deselects all checkboxes in the area with the list of SPIDs.

[Function buttons]

Button	Function
OK	Completes the settings of the SPID filter and closes this dialog box.
Cancel	Cancels the modification of the SPID filter and closes this dialog box.
Help	Displays the help of this dialog box.

Memory Mapping dialog box

This dialog box is used to set the memory mapping for each type of memory.

Caution When the selected microcontroller supports multi-core, this property displays the memory mapping status regarding a core (PE) by switching selection between the target cores (see "2.9 Select a Core (PE)").

Figure A.25 Memory Mapping Dialog Box [Full-spec emulator][E1][E20]

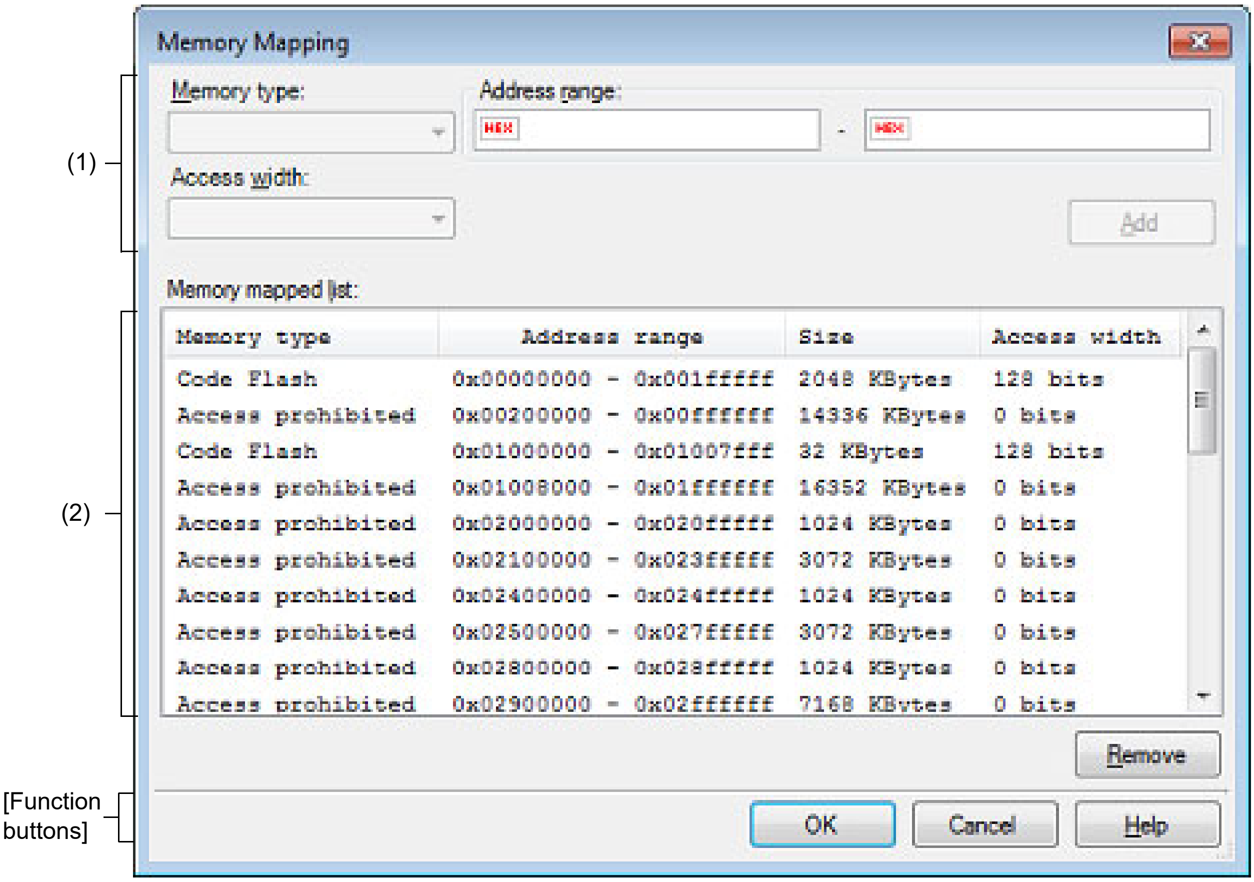
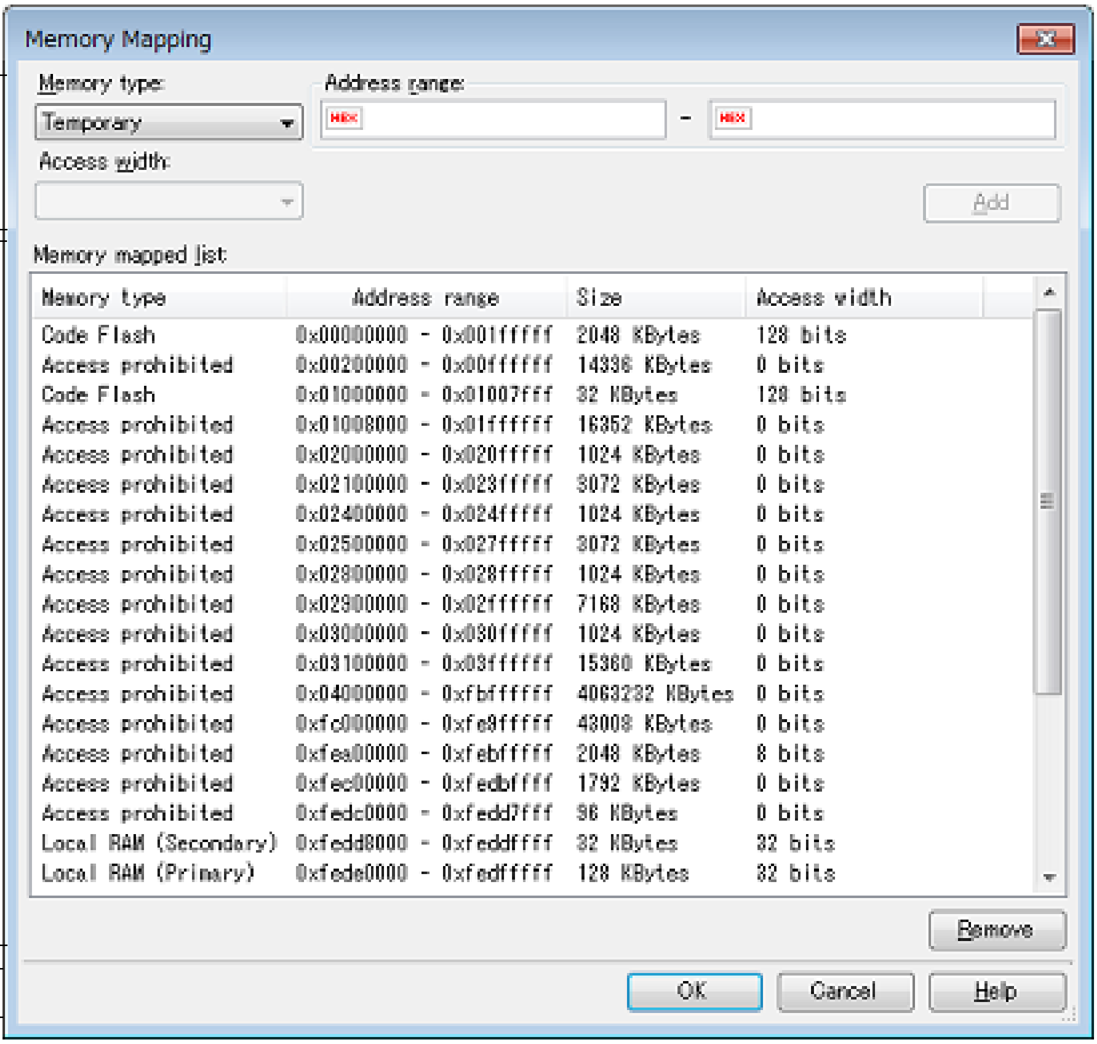


Figure A.26 Memory Mapping Dialog Box [Simulator]



(1)

(2)

[Function buttons]

This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Debug Tool Settings] tab of the Property panel, click the [...] button displayed by selecting one of the values of the [Memory mappings] property in the [Memory] category.

Caution This dialog box cannot be opened during execution of a program.

[Description of each area]

(1) Added memory mapping specification area
Specify the information for a memory mapping newly added.
This area is always invalid except for in the simulator.

(a) [Memory type] [Simulator]
Select the memory type for the memory mapping to be added from the following drop-down list (The item selected by default differs depending on the debug tool to use).

Temporary	This area is for testing and it can be accessed from the user program. It can be allocated to only space that overlaps with [Access prohibited].

Mapping attributes and their sizes that can be set are as follows:

Table A.11 Settable Mapping Attribute

Attribute	Debug Tool		
	Full-spec emulator	E1(LPD) E20(LPD)	Simulator
Temporary	-	-	✓

✓: Valid
-: Invalid

(b) [Address range]
Specify the start address and end address for the memory mapping to be added. Directly input a hexadecimal number into the text box for each.
In the case of the following settings, however, new memory mappings cannot be added (Clicking the [Add] button in this area causes an error message to be displayed).

- If the specified address range duplicates the memory area other than [Access prohibited] when [Temporary] is selected as the memory type

(c) [Access width] (except [Simulator])
This item is always invalid.

(d) Button

Button	Function
Add	Adds the content specified in this area to memory mapping. The added memory mapping is displayed in the [Memory mapped list] area. The changes will not take effect until the [OK] button is clicked.

(2) [Memory mapped list] area

(a) List display
Information about the memory mapping added in the Added memory mapping specification area and the microcontroller's internal memory mapping is displayed.
This area cannot be edited.

Memory type	Displays the following memory types. - Code Flash (*xxx*)[Note 1] - Local RAM (*xxx*)[Note 1] - Retention RAM (*xxx*)[Note 1] - Global RAM (*xxx*)[Note 1] - Cluster RAM (*xxx*)[Note 1] - Cluster Emulation RAM (*xxx*)[Note 1] - Global Emulation RAM (*xxx*)[Note 1] - Data Flash - Extend Data Area - Extend Data Area (*xxx*)[Note 1] - Block Protection Area - Configuration Setting Area - Security Setting Area - HBUS - CPU Peripheral (*xxx*)[Note 1] - PBUS - IBUS - FCU RAM - Emulation RAM - Video RAM - SDRAM - GTM RAM - Serial Flash - External Memory - Access prohibited - DFP RAM - DFP Peripheral - Temporary - Tag - Switch - Erase counter - MCS*x* RAM[Note 2]
Address range	Displays the address range as <*Start address*> - <*End address*>. Display is fixed as "0x"-prefixed hexadecimal numbers.
Size	Displays size as a decimal number (unit: bytes/Kbytes[Note 3]).
Access width	Displays the access width (unit: bits).

Note 1. The following additional information is entered to *xxx*.

- Bank information (e.g. BankA, BankB)
- PE number (e.g. PE1, PE3)
- Unique names, such as Self and PCU (e.g. PCU, Self, Primary, Secondary)

Note 2. The selected MCS number is entered as *x*.

Note 3. Only in the case of multiple of 1024, displays in kilobyte units.

(b) Button

Button	Function
Remove	Deletes the memory mapping selected in this area. The memory area that can be deleted is the memory mapping added by the user (the microcontroller's internal memory mapping cannot be deleted).

[Function buttons]

Button	Function
OK	Sets the currently specified memory mapping to the debug tool and closes this dialog box.
Cancel	Cancels memory mapping changes and closes this dialog box.
Help	Displays the help for this dialog box.

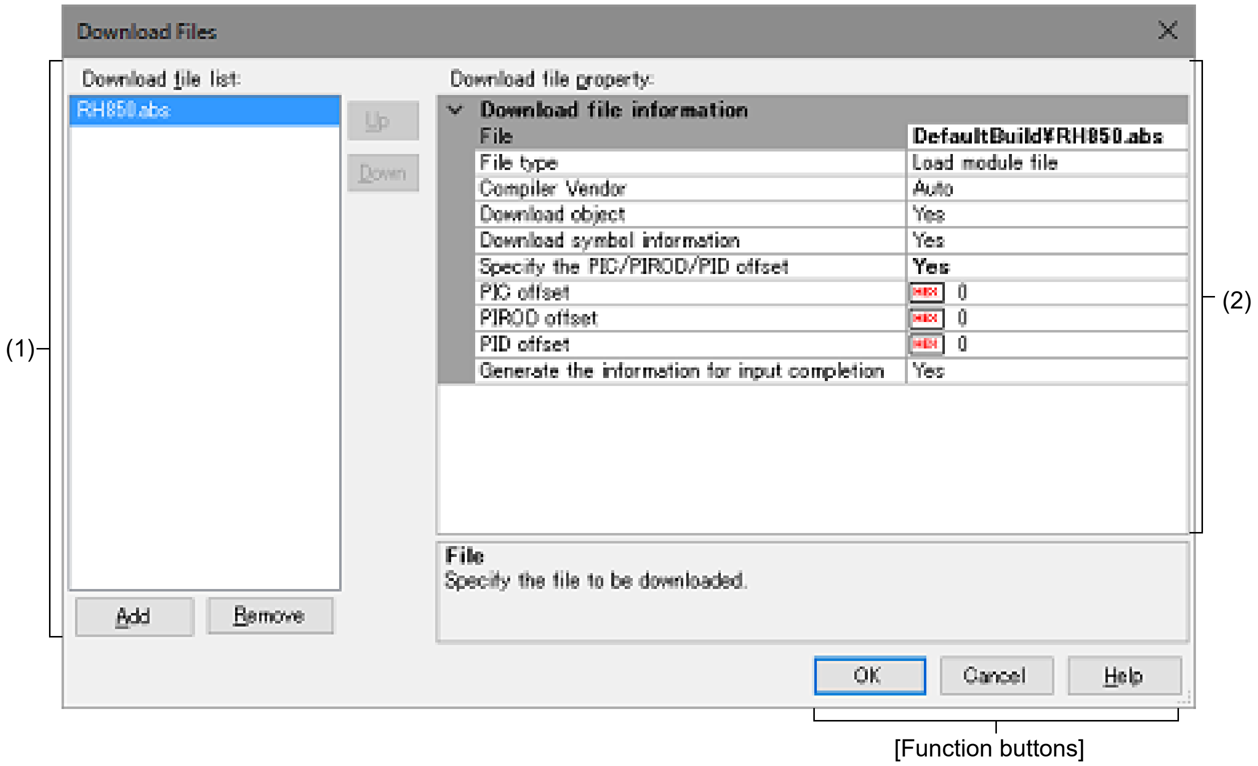
Download Files dialog box

This dialog box is used to select files for downloading and configure download conditions (see "2.6 Download/Upload Programs").

Note that files specified as build targets in a project (main project or sub-project) are automatically registered as download targets (they can be unregistered).

Caution This dialog box cannot be opened during execution of a program.

Figure A.27 Download Files Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Download File Settings] tab of the Property panel, click the [...] button displayed by selecting the [Download files] property in the [Download] category.

[Description of each area]

(1) [Download file list] area

(a) List display

Displays a list of files to download. The names of files specified as build targets in a project (main project or sub-project) are displayed by default (they can be removed).

Files are downloaded in the order that they are displayed here.

To add a new file to be downloaded, click the [Add] button in this area, then in the [Download file property] area, specify the download conditions of the file to add.

(b) Button

Button	Function
Up	Moves the selected file up one row in the list. Clicking this for the top file in the list has no effect.
Down	Moves the selected file down one row in the list. Clicking this for the bottom file in the list has no effect.
Add	Adds an empty item "-" to the list, and selects it. Specify the download conditions of the file to add in the [Download file property] area. Note that this button will be disabled if 20 files have already been registered.
Remove	Removes the selected file from the list. Note, however, that this button is disabled if the selected file is a project build target.

Remark 1. By hovering the mouse cursor over a file name, the pass information of the file is pop-up displayed.

Remark 2. By dragging a file name with the mouse, the display order in the list can be changed.
Note, however, that the order of a project build target cannot be changed.

(2) [Download file property] area

(a) [Download file information]
This area is used to display or edit the download conditions of the file selected in the [Download file list] area.
It can also be used to specify the download conditions of new download files added via the [Add] button.

File	Specify the name of the file to download.		
	Default	*File name* (but it will be blank for newly added files)	
	Modifying	Directly enter from the keyboard, or specify with the Select Download File dialog box opened by clicking the [...] button[Note 1] appears at right by selecting this item.	
	Available values	See "Table 2.4 Downloadable File Formats" Up to 259 characters	
File type	Select the type of the file to download.		
	Default	Load module file	
	Modifying	Select from the drop-down list.	
	Available values	Load module file	Specifies a load module file.
		Hex file	Specifies an Intel HEX file.
		S-record file	Specifies a Motorola S-record file.
		Binary data file	Specifies an binary file.
Compiler Vendor	Specify the vendor of the compiler that was used to create a load module file. This property setting is used to handle the information that was output in the load module file and is specific to the compiler's vendor.		
	Default	Auto	
	Modifying	Select from the drop-down list.	
	Available values	Auto	Specify when having the compiler determined automatically.
		Green Hills Software	Specify when using a GHS compiler.

Offset	Specify the offset from the address at which the file's download is to start. Note that this item appears only when [File type] is set to [Hex file] or [S record file].		
	Default	0	
	Modifying	Directly enter from the keyboard.	
	Available values	0x0 to 0xFFFFFFFF in hexadecimal number	
Start address	Specify the address at which to start the file's download. Note that this item appears only when [File type] is set to [Binary file].		
	Default	0	
	Modifying	Directly enter from the keyboard.	
	Available values	0x0 to 0xFFFFFFFF in hexadecimal number	
Download object	Select whether to download the object information from the specified file. Note that this item appears only when [File type] is set to [Load module file].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloads object information.
		No	Does not download object information.
Download symbol information	Select whether to download the symbol information from the specified file[Note 2]. Note that this item appears only when [File type] is set to [Load module file].		
	Default	Yes	
	Modifying	Select from the drop-down list.	
	Available values	Yes	Downloads symbol information.
		No	Does not download symbol information.
Specify the PIC/ PIRDD/PID offset	Specify whether to change the positions of PIC (Position Independent Code), PIRDD (Position Independent Read Only Data) 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", "PIROD Offset" and "PID Offset" will appear as sub-items.		
	Default	No	
	How to change	Select from the drop-down list.	
	Specifiable value	Yes	PIC/PIROD/PID offset is specified[Note 3].
		No	PIC/PIROD/PID offset is not specified.
PIC Offset	Input the offset values from the start address of the program section specified at the time of load module creation. For example, if 1000 is specified for this item when a load module for which the program section starts at address 1000 is to be downloaded, the section will be downloaded to address 2000.		
	Default	0	
	How to change	Enter directly from the keyboard.	
	Specifiable value	Hex number between 0x0 and 0xFFFFFFFF	

<table>
<tr><td rowspan="4">PIROD Offset</td><td colspan="3">Input the offset values from the start address of the ROM data section specified at the time of load module creation.
For example, if 1000 is specified for this item when a load module for which the ROM data section starts at address 1000 is to be downloaded, the section will be downloaded to address 2000.</td></tr>
<tr><td>Default</td><td colspan="2">0</td></tr>
<tr><td>How to change</td><td colspan="2">Enter directly from the keyboard.</td></tr>
<tr><td>Specifiable value</td><td colspan="2">Hex number between 0x0 and 0xFFFFFFFF</td></tr>
<tr><td rowspan="4">PID Offset</td><td colspan="3">Input the offset values from the start address of the RAM data section specified at the time of load module creation.
For example, if 1000 is specified for this item when a load module for which the RAM data section starts at address 1000 is to be downloaded, the section will be downloaded to address 2000.</td></tr>
<tr><td>Default</td><td colspan="2">0</td></tr>
<tr><td>How to change</td><td colspan="2">Enter directly from the keyboard.</td></tr>
<tr><td>Specifiable value</td><td colspan="2">Hex number between 0x0 and 0xFFFFFFFF</td></tr>
<tr><td rowspan="5">Generate the information for input completion</td><td colspan="3">Select whether to generate the information for the Symbol name completion function when downloading[Note 4].
Note that this item appears only when [File type] is set to [Load module file].</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Generates the information for the symbol name completion function. (i.e. uses the symbol name completion function.)</td></tr>
<tr><td>No</td><td>Does not generate the information for the symbol name completion function. (i.e. does not use the symbol name completion function.)</td></tr>
</table>

Note 1. When a file specified as build target in the project is selected in the [Download file list] area, or when the program is executing, the [...] button does not appear.

Note 2. If the symbol information have not been downloaded, the source level debugging 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 "2.8 Usage of PIC/PID Function").

Note 4. When [Yes] is selected, the time taken for downloading and the memory usage on the host machine will increase. We recommend selecting [No] in this item if you do not intend to use the symbol name completion function.

[Function buttons]

Button	Function
OK	Finishes configuring the download files, and closes this dialog box.
Cancel	Cancels any changes to the download files, and closes this dialog box.
Help	Displays the help for this dialog box.

Flash Options Setting dialog box

This dialog box is used to configure options for the flash memory incorporated in the microcontroller.

Caution 1. [Full-spec emulator][E1][E20]
This dialog box appears only when connecting to the debug tool.
[Simulator]
This dialog box appears only when disconnecting from the debug tool.

Caution 2. [Full-spec emulator][E1][E20]
CPU reset may be generated automatically when you click the [Write] button after changing the configuration of this dialog box.

Figure A.28 Flash Options Setting Dialog Box [Full-spec emulator][E1][E20]

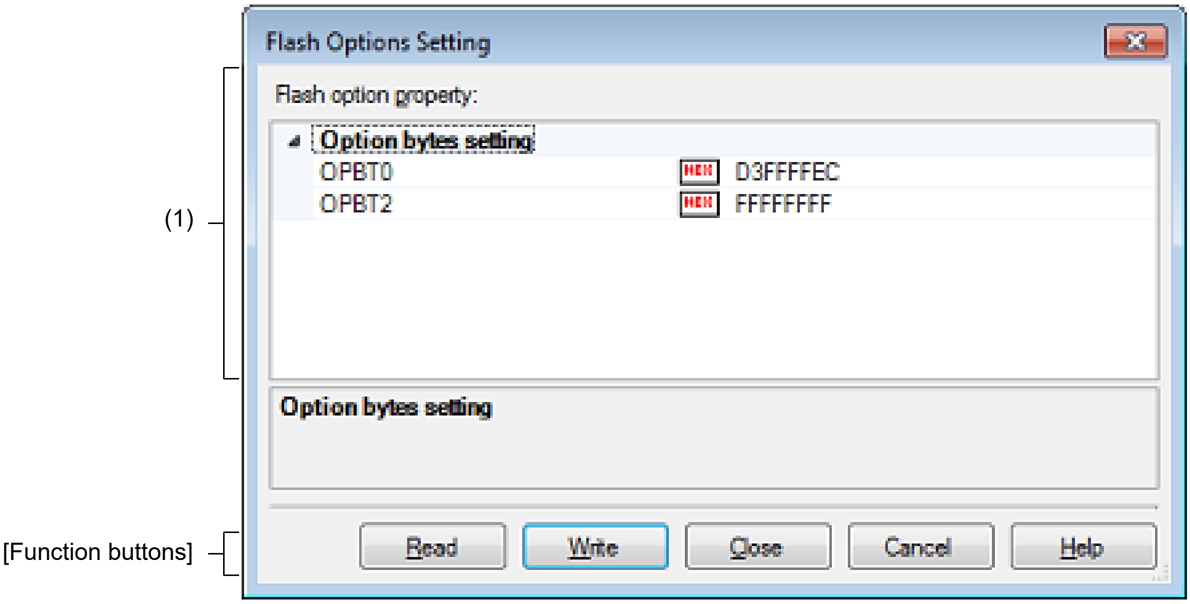
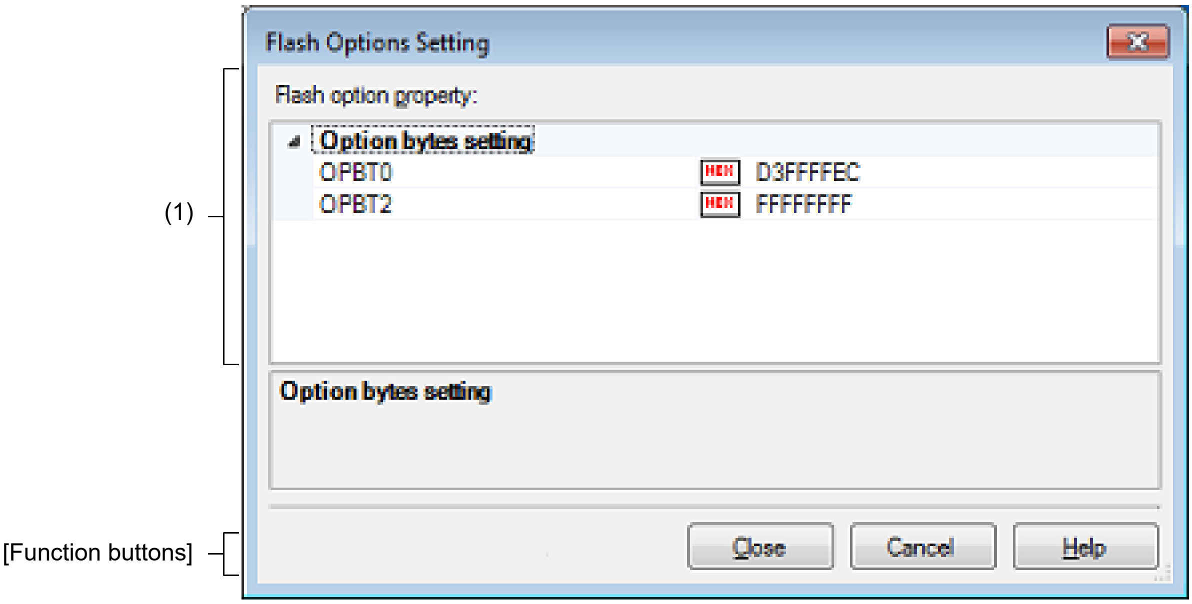


Figure A.29 Flash Options Setting Dialog Box [Simulator]



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Flash Options Settings] tab tab of the Property panel, click the [...] button displayed by selecting the [Flash options] property in the [Flash Options] category.

[Description of each area]

(1) [Flash option property] area

(a) [Option Bytes Setting]
You can configure the option bytes for the flash memory.

OPBT*0 - 15*	Specify the option bytes.	
	Default	The initial value stored in the device file.
	Modifying	Directly enter from the keyboard.
	Available values	0x0 to 0xFFFFFFFF in hexadecimal number

Caution The number of option bytes (OPB*0 - 15*) to be displayed differs with the selected microcontroller (the bytes for some numbers might actually be unused).

[Function buttons]

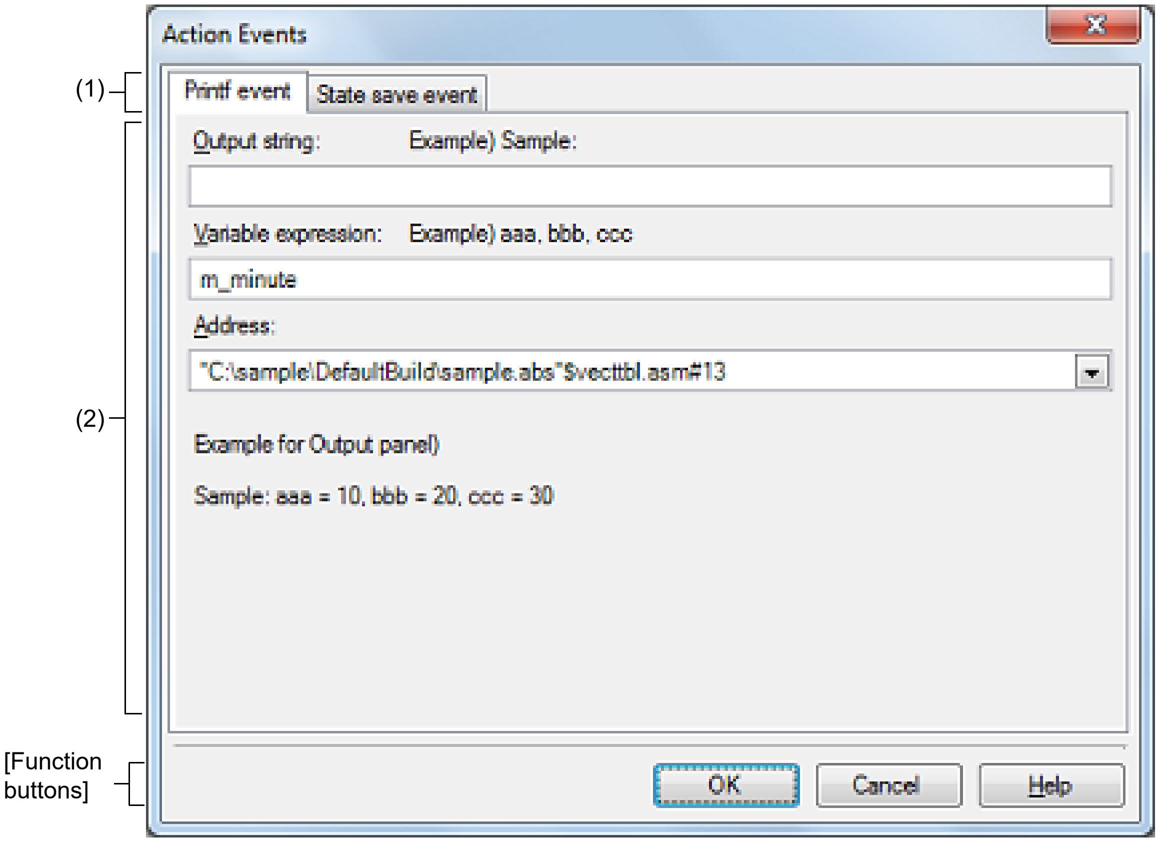
Button	Function
Read [Full-spec emulator][E1][E20]	Reads the values currently specified in the debug tool, and reflects them in this dialog box.
Write [Full-spec emulator][E1][E20]	Writes the currently set values in this dialog box to the debug tool, and reflects them in the project. Then, closes this dialog box.
Close	Specifies the currently set values in this dialog box to the project and closes this dialog box.
Cancel	Closes this dialog box without setting.
Help	Displays the help for this dialog box.

Action Events dialog box

This dialog box is used to configure action events (see "2.18 Set an Action into Programs").
This dialog box appears only when connected to the debug tool.

Caution Also see "2.19.7 Notes for setting events" for details on Printf events (e.g. limits on the number of enabled events).

Figure A.30 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 an action event, then select [Register Action Event...] from the context menu.
- On the Events panel, select an action event, then select [Edit Condition...] from the context menu.

[Description of each area]

(1) Tab selection area
Select a tab to switch the type of an action event to be set.
This dialog box has the following two tabs.

- [Printf event] tab
- [State save event] tab

Caution If this dialog box is opened by selecting [Edit Condition...] from the context menu, this area does not appear.

(2) Event condition setting area
Use this area to configure detailed condition of an action event.
For details on how to setup an action event, see the section explaining the corresponding tab.

[Function buttons]

Button	Function
OK	Finishes configuring the action event, and sets it at the position specified in this dialog box.
Cancel	Cancels the action event settings and closes this dialog box.
Help	Displays the help for this dialog box.

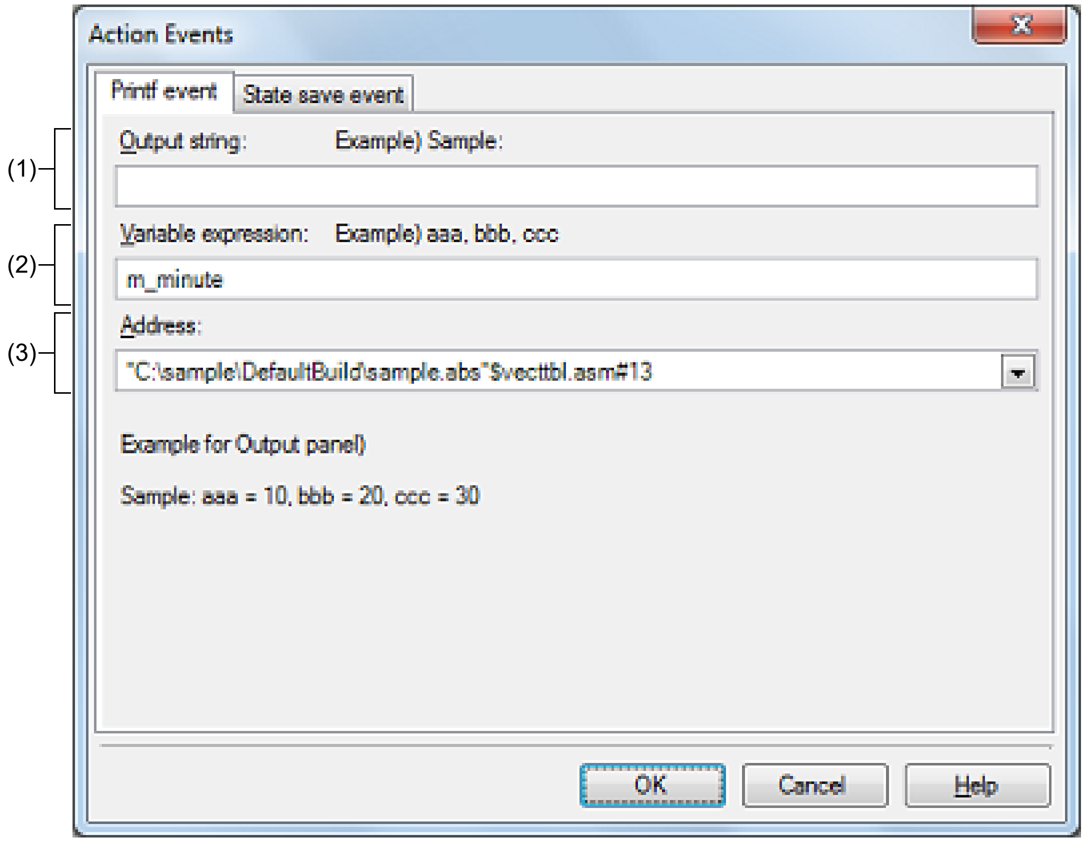
[Printf event] tab

This tab is used to configure Printf events as action events (see "2.18 Set an Action into Programs").

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.

This dialog box appears only when connected to the debug tool.

Figure A.31 Action Events Dialog Box: [Printf event] Tab



This section describes the following.

- [How to open]
- [Description of each area]

[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 a Printf event, then select [Edit Condition...] from the context menu.

[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 opens 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.12 Relationship between Variable Expressions and Output Value (Printf Event)

Variable Expression	Output Value
Variable name of C language	Value of C language variable
Variable expression [*Variable expression*]	Element of array
Variable expression.Member name	Member of structure/union
Variable expression -> Member name	Member of structure/union that pointer designates
**Variable expression*	Value of pointer variable
&*Variable expression*	Location address
CPU register name	Value of the CPU register
I/O register name	I/O register value
Label name[Note], EQU symbol name[Note] and [immediate address]	Values of label, EQU symbol and immediate address

Note 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 A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (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 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.

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (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.32 Output Result Format of Printf Event

Specified-characters Variable-expression-1 = Value-1, Variable-expression-2 = Value-2, ...

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 the default notation (see "Table A.7 Display Format of Watch-Expressions (Default)") according to the type of the variable (note, however, that "?" will be displayed if the specified variable expression cannot be obtained). Moreover, the value in hexadecimal number enclosing with "()" is also displayed (note, however, that "-" will be displayed if the value cannot be displayed in that notation).

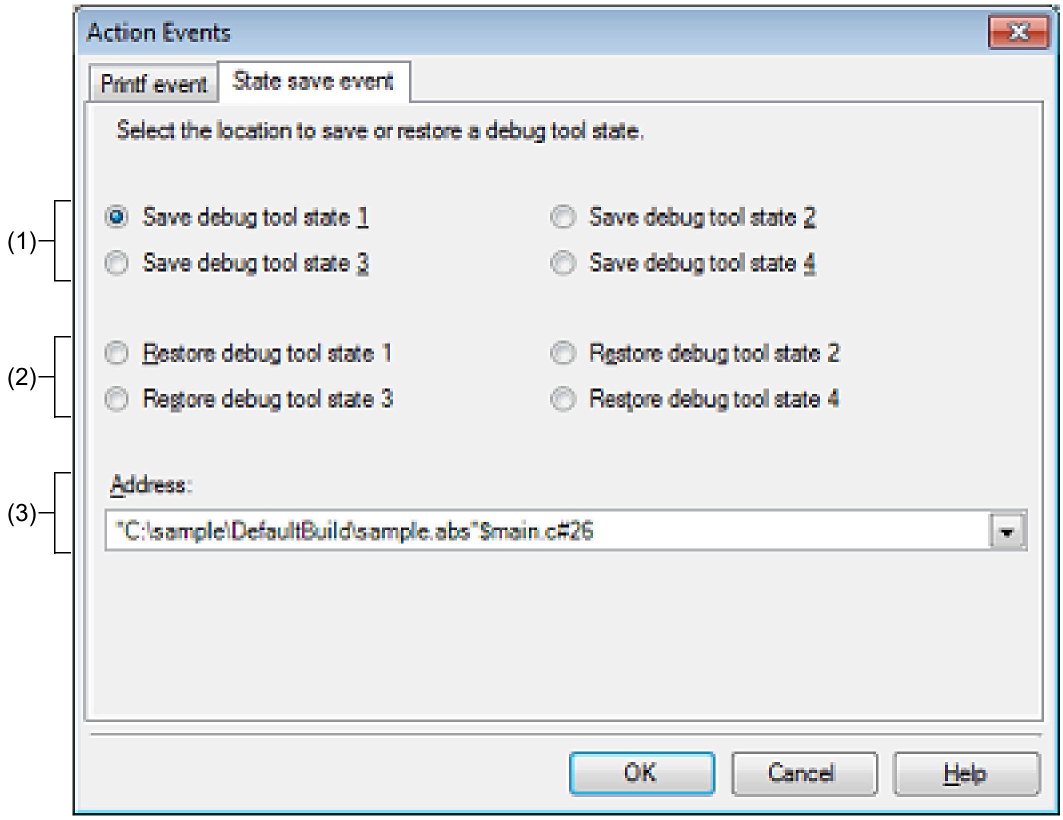
[Function buttons]

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 settings 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.33 Action Events Dialog Box: [State save event] Tab



This section describes the following.

- [How to open]
- [Description of each area]

[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 a State save event, then select [Edit Condition...] from the context menu.

[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.

For details on the features and usage, see “Saving and Restoring the States of Debug Tools”.

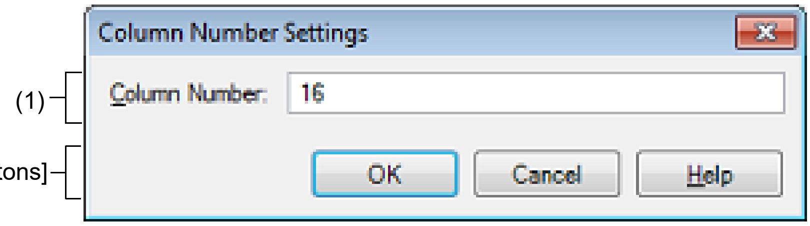
[Function buttons]

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.

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.34 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).

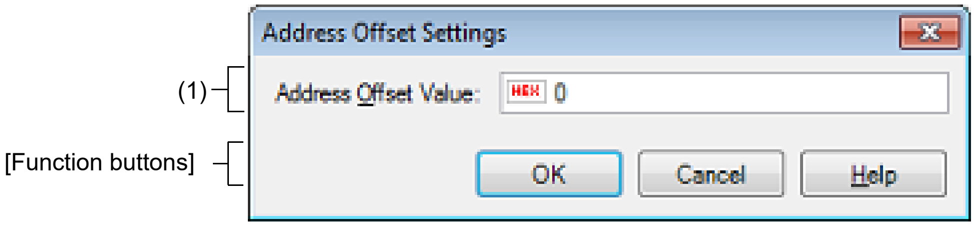
[Function buttons]

Button	Function
OK	Displays memory values in the specified number of columns.
Cancel	Cancels the settings and 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.35 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

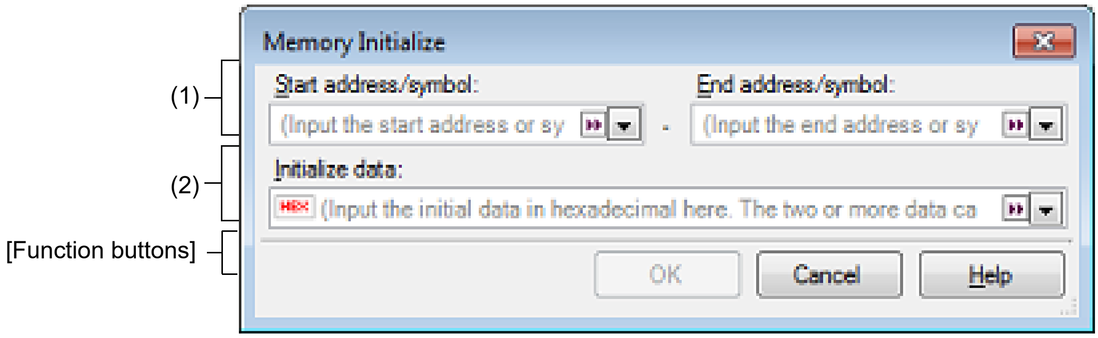
[Function buttons]

Button	Function
OK	Displays memory addresses with the specified offset value.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help for this dialog box.

Memory Initialize dialog box

This dialog box is used to initialize memory (see "2.12.1.6 Modify the memory contents in batch (initialize)").
The memory area in the specified address range is repeatedly overwritten with the specified initial data pattern.

Figure A.36 Memory Initialize Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Memory panel, select [Fill…] from the context menu.

[Description of each area]

(1) Range specification area
Specify the range of memory address to initialize via the [Start address/symbol] and [End address/symbol]. 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).
The results of calculating the address expressions you have entered are treated as start and end addresses, respectively.
Note that address values greater than the microcontroller address space cannot be specified.

Caution You cannot specify the range of address aligned across the different endian area.

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

(2) [Initialize data] area
Specify the initial value(s) with which to overwrite the memory.
You can either type the initial value into the text box directly in hexadecimal number (the value need not start with "0x"), or select one from the input history via the drop-down list (up to 10 items).
You can specify more than one initial value. Specify up to 16 values of up to 4 bytes (8 characters) each, separated by spaces.
Each initial value is parsed from the end of the string, with each two characters interpreted as a byte.
If the string has an odd number of characters, then the first character is interpreted as one byte.
Note that if a initial value consists of more than one byte, then the target memory is overwritten with the value converted into an array of bytes of the specified address range's endian, as follows:

Input Character Strings (Initial Value)	How Data is Overwritten (in Bytes)	
	Little Endian	Big Endian
1	01	01
0 12	00 12	00 12

Input Character Strings (Initial Value)	How Data is Overwritten (in Bytes)	
	Little Endian	Big Endian
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

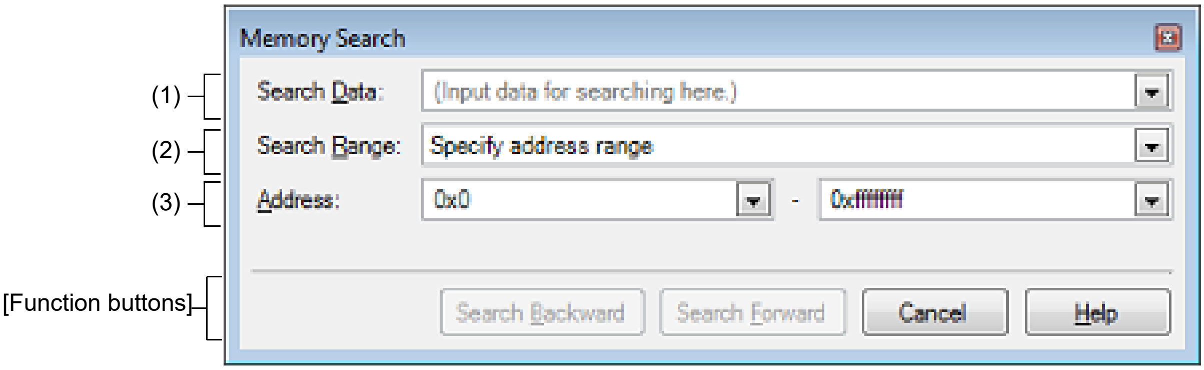
[Function buttons]

Button	Function
OK	The memory area in the specified address range is repeatedly overwritten with the specified initial data pattern. If the end address is reached in the middle of the pattern, then writing ends at that point.
Cancel	Cancels the memory initialization and closes this dialog box.
Help	Displays the help for this dialog box.

Memory Search dialog box

This dialog box is used to search memory (see "2.12.1.5 Search the memory contents").
Search in either the Memory value area or Character strings area where the caret was located in the Memory panel immediately before this dialog box opened.

Figure A.37 Memory Search Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Memory panel, select [Find…] from the context menu.

[Description of each area]

(1) [Search Data] area
Specify data to search.
You can either type a value directly into the text box (up to 256 bytes), or select one from the input history via the drop-down list (up to 10 items).
If the search is performed in the Memory value area of the Memory panel, the value must be entered in the same display format (notation and size) as that area.
If the search is performed in the Character strings area, then the target of the search must be a string. The specified string is converted into the encoding format displayed in that area, and searched for.
If a memory value was selected immediately prior to opening this dialog box, then that value will appear as default.

(2) [Search Range] area
Select the range to search from the following drop-down list.

Specify address range	Searches in the address range specified in the [Address] area.
Memory mapping	Searches within the selected memory mapping range. This list item displays the memory mappings set in the Memory Mapping dialog box. Display format: *<memory type> <address range> <size>*

(3) [Address] area
This item is only enabled if [Specify address range] is selected in the [Search Range] area.
Specify the range of memory address to search via the start and end addresses. 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).
The results of calculating the address expressions you have entered are treated as start and end addresses, respectively.
Note, however, that the largest address that can be searched is the maximum address of the program space (0x03FFFFFF) (the mirror area cannot be searched).

In addition, an address value greater than the value expressed within 32 bits cannot be specified.

Remark 1. A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

Remark 2. If the start address field is left blank, it is treated as if "0x0" were specified.

Remark 3. If the end address field is left blank, then it is treated as if the maximum value in the microcontroller's address space were specified.

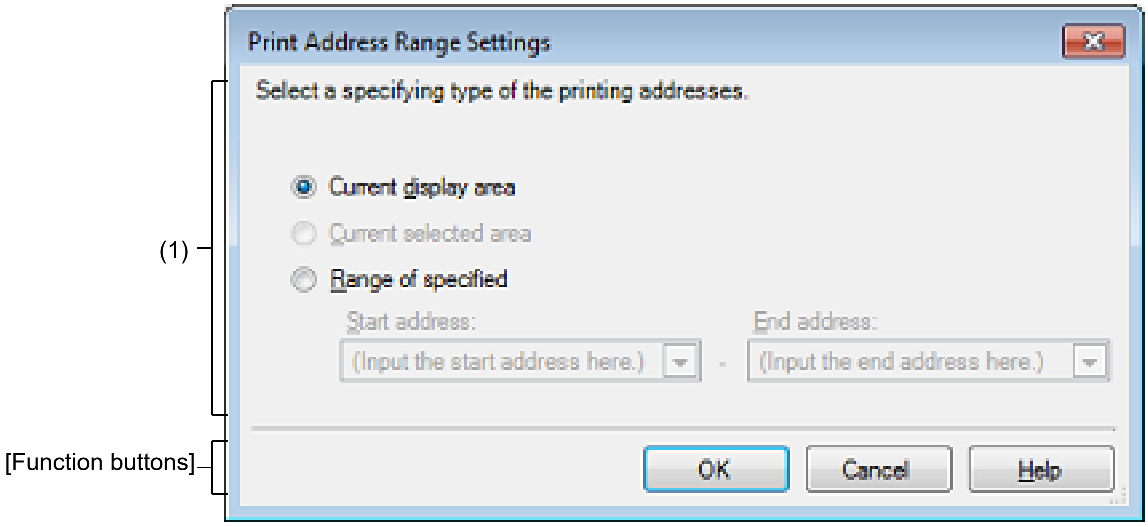
[Function buttons]

Button	Function
Search Backward	Searches upward within the range specified in the [Address] area or the [Search Range] area. The location found by the search is selected in the Memory panel. Note that if an illegal value is specified or while the program is being executed, a message will appear, and the memory search will not be performed. If focus moves to this dialog box while the memory panel is hidden or another panel has focus, then this button will be disabled.
Search Forward	Searches downward within the range specified in the [Address] area or the [Search Range] area. The location found by the search is selected in the Memory panel. Note that if an illegal value is specified or while the program is being executed, a message will appear, and the memory search will not be performed. If focus moves to this dialog box while the memory panel is hidden or another panel has focus, then this button will be disabled.
Cancel	Cancels the memory search and closes this dialog box.
Help	Displays the help for this dialog box.

Print Address Range Settings dialog box

This dialog box is used to specify the address range to print the contents of the Disassemble panel.

Figure A.38 Print Address Range Settings Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Disassemble panel, select [Print...] from the [File] menu.

[Description of each area]

(1) Range specification area
Select a range to print from the following option buttons.

(a) [Current display area] (default)
Prints only the contents of the Disassemble panel currently being displayed.

(b) [Current selected area]
Prints only the range currently being selected in the Disassemble panel.
Note, however, that this option button will be disabled when nothing is selected in the Disassemble panel.

(c) [Range of specified]
Specify the range of address to print via [Start address] and [End address]. You can either type address expressions directly into the text boxes (up to 1024 characters), or select them from the input history via the drop-down list (up to 10 items).

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

[Function buttons]

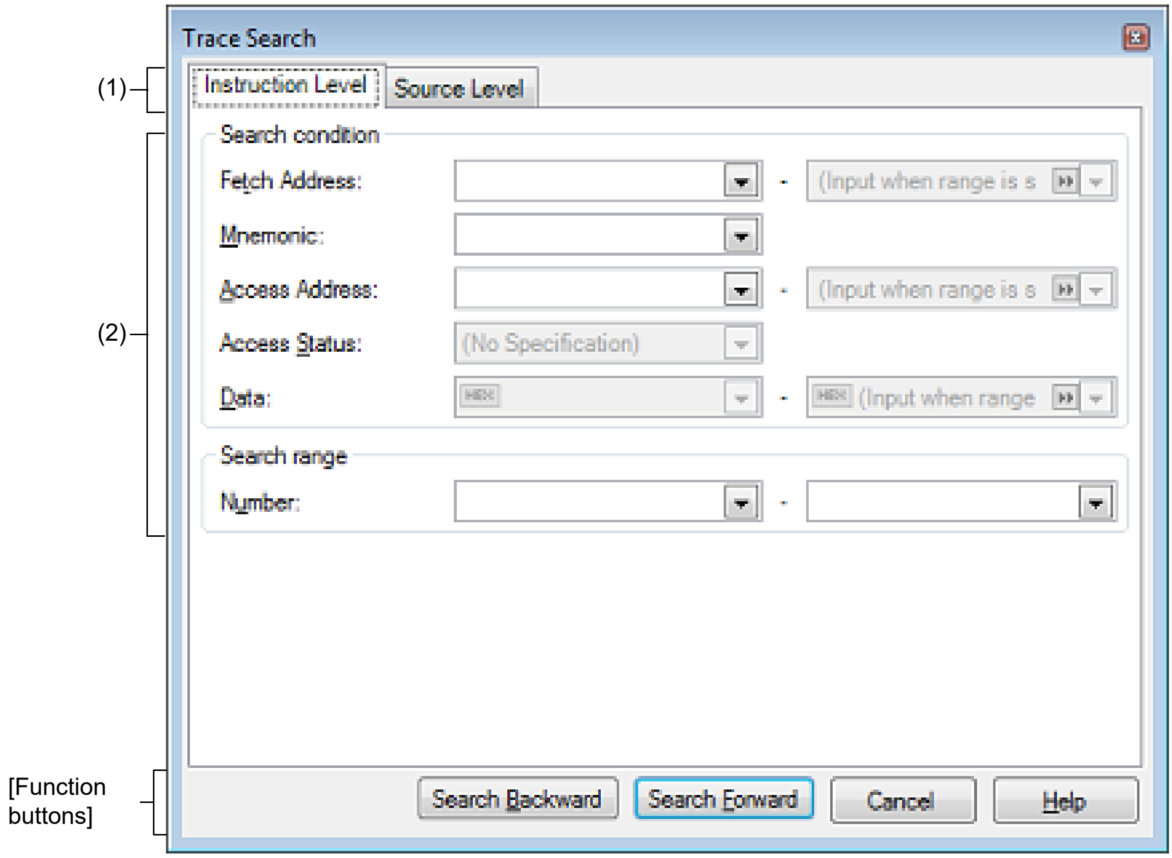
Button	Function
OK	Closes this dialog box and opens the Windows dialog box to print the contents of the specified range of the Disassemble panel.

Button	Function
Cancel	Cancels the range specification and closes this dialog box.
Help	Displays the help for this dialog box.

Trace Search dialog box

This dialog box is used to search for trace data (see "2.14.8 Search the trace data").
The search can be performed at the instruction or source level.

Figure A.39 Trace Search Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Trace panel, select button on the toolbar.
- On the Trace panel, select [Find...] from the context menu.

[Description of each area]

(1) Tab selection area
Select a tab to switch the level of the search.
This dialog box has the following two tabs.

- [Instruction Level] tab
- [Source Level] tab

(2) Search parameter setup area
Use this area to configure detailed search parameters.

For details on the window elements and how to configure the parameters for a particular tab, see the section for the tab in question.

[Function buttons]

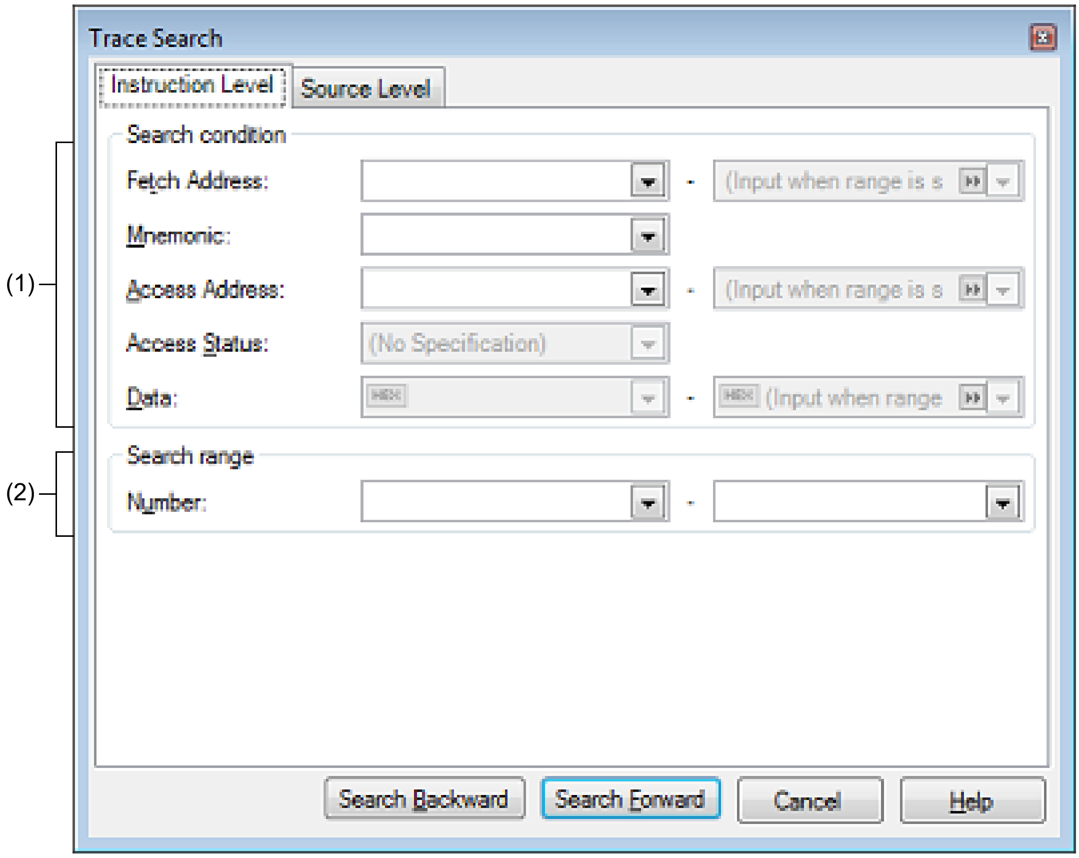
Button	Function
Search Backward	Searches upward (in the direction of larger to smaller numbers) within the specified range. Search matches are selected in the Trace panel. Note that if an illegal value is specified or while the program is being executed, a message will appear, and the trace data search will not be performed. If focus moves to this dialog box while the Trace panel is hidden or another panel has focus, then this button will be disabled.
Search Forward	Searches forward (in the direction of smaller to larger numbers) within the specified range. Search matches are selected in the Trace panel. Note that if an illegal value is specified or while the program is being executed, a message will appear, and the trace data search will not be performed. If focus moves to this dialog box while the Trace panel is hidden or another panel has focus, then this button will be disabled.
Cancel	Cancels the trace data search and closes this dialog box.
Help	Displays the help for this dialog box.

[Instruction Level] tab

This tab is used to search for the acquired trace data at the instruction level.

Caution If the Trace panel is set to Source display mode, then performing an instruction level search via this tab will not perform the target search correctly. In order to perform an instruction level search, set the mode to Mixed display mode or Disassemble display mode.

Figure A.40 Trace Search Dialog Box: [Instruction Level] Tab



This section describes the following.

- [How to open]
- [Description of each area]

[How to open]

- On the Trace panel, select  button on the toolbar.
- On the Trace panel, select [Find…] from the context menu.

[Description of each area]

(1) [Search condition] area

(a) [Fetch Address]
Specify the fetch address if it is a required search parameter.
You can either type address expressions directly into the text boxes, or select them from the input history via the drop-down lists (up to 10 items).
The fetch address can also be specified as a range. In this case, specify a range by specifying address expressions in both the left and right text boxes.
If the right-hand text box is blank or contains the text [(Input value when range is specified)], then the fixed address specified in the left-hand text box will be searched.

Note that if an address value greater than the microcontroller address space is specified, the upper address value is masked.
In addition, an address value greater than the value expressed within 32 bits cannot be specified.

(b) [Mnemonic]
Specify the mnemonic if it is a required search parameter.
The specified character strings in this area are searched within the [Source/Disassemble] area of the Trace panel.
You can either type a mnemonic directly into the text boxes, or select one from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and partial matches are also allowed.

(c) [Access Address]
Specify the access address if it is a required search parameter.
You can either type address expressions directly into the text boxes, or select them from the input history via the drop-down lists (up to 10 items).
The access address can also be specified as a range. In this case, specify a range by specifying address expressions in both the left and right text boxes.
If the right-hand text box is blank or contains the text [(Input value when range is specified)], then the fixed address specified in the left-hand text box will be searched.
Note that if an address value greater than the microcontroller address space is specified, the upper address value is masked.
In addition, an address value greater than the value expressed within 32 bits cannot be specified.

(d) [Access Status]
This item is only enabled if a value for [Access Address] is specified.
Select the access type from the following drop-down list.

(No Specification)	Does not limit access types.
Read/Write	Searches the location where a read or write access occurred.
Read	Searches the location where a read access occurred.
Write	Searches the location where a write access occurred.
Vector Read	Searches the location where a vector read access occurred.
DMA	This item is invalid.

(e) [Data]
This item is only enabled if a value for [Access Address] is specified.
Specify the access data.
You can either type the data directly into the text boxes (in hexadecimal number), or select it from the input history via the drop-down list (up to 10 items).
The data can also be specified as a range. In this case, specify a range by specifying data in both the left and right text boxes.
If the right-hand text box is blank or contains the text [(Input value when range is specified)], then the fixed data specified in the left-hand text box will be searched.

(2) [Search range] area

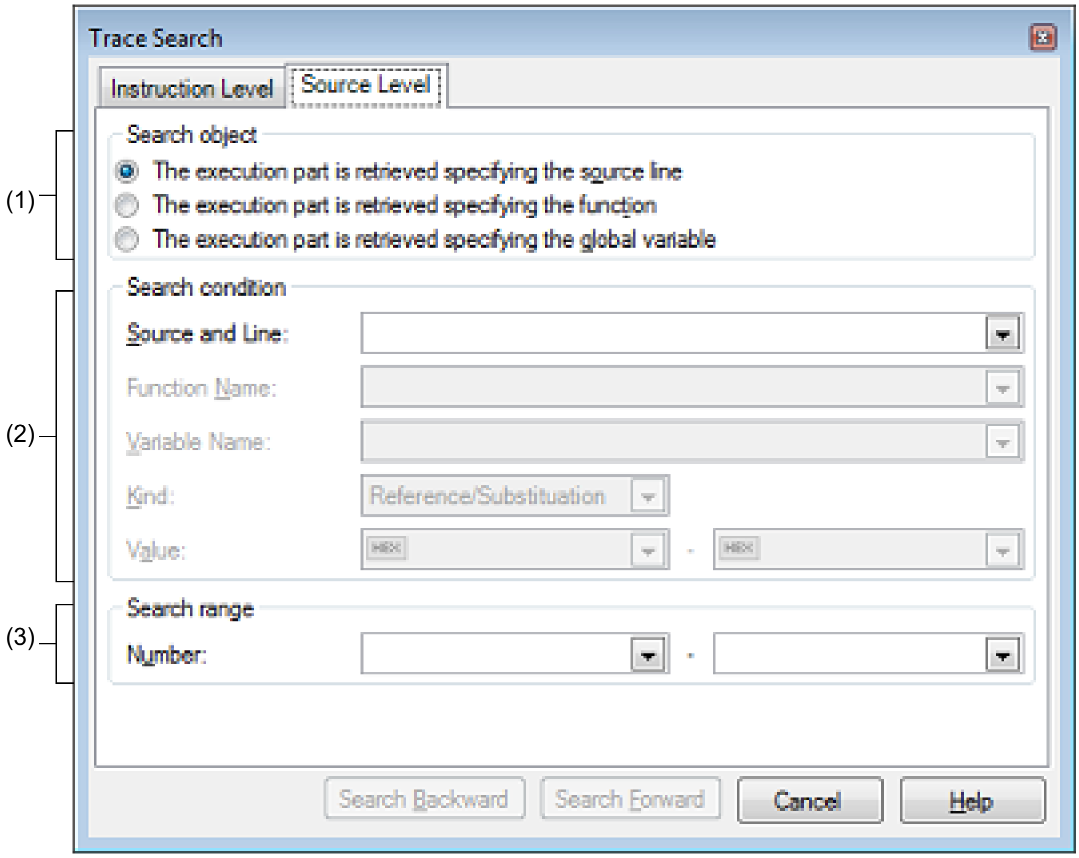
(a) [Number]
Specify the range within the trace data to search via the number displayed in the [Number] area of the Trace panel.
Specify the starting number in the left text box, and the ending number in the right text box ("0" to "last number" are specified by default).
You can either type the numbers directly into the text boxes (in base-10 format), or select them from the input history via the drop-down lists (up to 10 items).
If the left-hand text box is left blank, it is treated as if "0" were specified.
If the right-hand text box is left blank, it is treated as if the last number were specified.

[Source Level] tab

This tab is used to search for the acquired trace data at the source level.

Caution If the Trace panel is set to Disassemble display mode, then performing an source level search via this tab will not perform the target search correctly. In order to perform an source level search, set the mode to Mixed display mode or Source display mode.

Figure A.41 Trace Search Dialog Box: [Source Level] Tab



This section describes the following.

- [How to open]
- [Description of each area]

[How to open]

- On the Trace panel, select button on the toolbar.
- On the Trace panel, select [Find...] from the context menu.

[Description of each area]

(1) [Search object] area
Select the search object from the following option buttons.

The execution part is retrieved specifying the source line	Finds the execution location in the specified source line (default). Only [Source and Line] will be enabled as a search parameter.
The execution part is retrieved specifying the function	Finds the execution location in the specified function. Only [Function Name] will be enabled as a search parameter.

The execution part is retrieved specify-ing the global variable	Finds the location at which the specified global variable was accessed. Only [Variable Name], [Kind] and [Value] will be enabled as a search parameters.

(2) [Search condition] area

(a) [Source and Line]
This item is only enabled if [The execution part is retrieved specifying the source line] is selected.
The specified character strings in this area are searched within the [Line Number/Address] area of the Trace panel.
You can either type the character strings of the source line to be find directly into the text box, or select them from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and only complete matches are retrieved.

Example 1. main.c#40

Example 2. main.c

Example 3. main

(b) [Function Name]
This item is only enabled if [The execution part is retrieved specifying the function] is selected.
You can either type the function name to be find directly into the text box, or select it from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and only complete matches are retrieved.

(c) [Variable Name]
This item is only enabled if [The execution part is retrieved specifying the global variable] is selected.
You can either type the variable name to be find directly into the text box, or select it from the input history via the drop-down list (up to 10 items).
Searches are case-insensitive, and only complete matches are retrieved.

(d) [Kind]
This item is only enabled if [The execution part is retrieved specifying the global variable] is selected.
Select the access type ([Reference/Substitution], [Reference], or [Substitution]) from the drop-down list.

(e) [Value]
This item is only enabled if [The execution part is retrieved specifying the global variable] is selected.
Specify the accessed variable value in hexadecimal number.
You can either type a variable value directly into the text box, or select one from the input history via the drop-down list (up to 10 items).
The variable value can also be specified as a range. In this case, specify a range by specifying variable values in both the left and right text boxes.
If the right-hand text box is blank, then access locations with the fixed variable values specified in the left-hand text box will be searched for.

(3) [Search range] area

(a) [Number]
Specify the range within the trace data to search via the number displayed in the [Number] area of the Trace panel.
Specify the starting number in the left text box, and the ending number in the right text box ("0" to "last number" are specified by default).
You can either type the numbers directly into the text boxes (in base-10 format), or select them from the input history via the drop-down lists (up to 10 items).
If the left-hand text box is left blank, it is treated as if "0" were specified.
If the right-hand text box is left blank, it is treated as if the last number were specified.

Detail dialog box (for execution events)

This dialog box is used to display and modify detailed information on an execution-related event selected in the Events panel.

Figure A.42 Detail Dialog Box (for Execution Events) **[Full-spec emulator][E1][E20]**

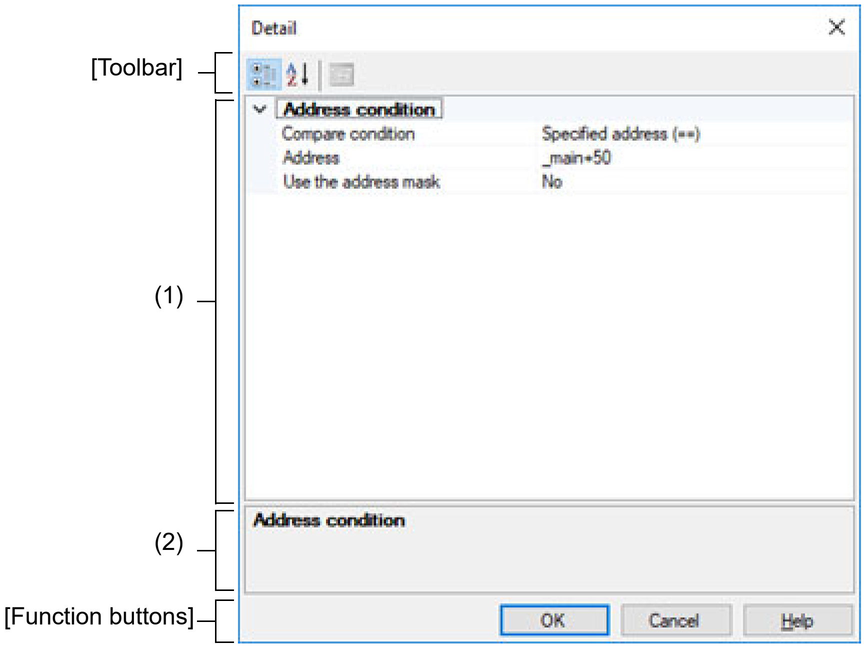
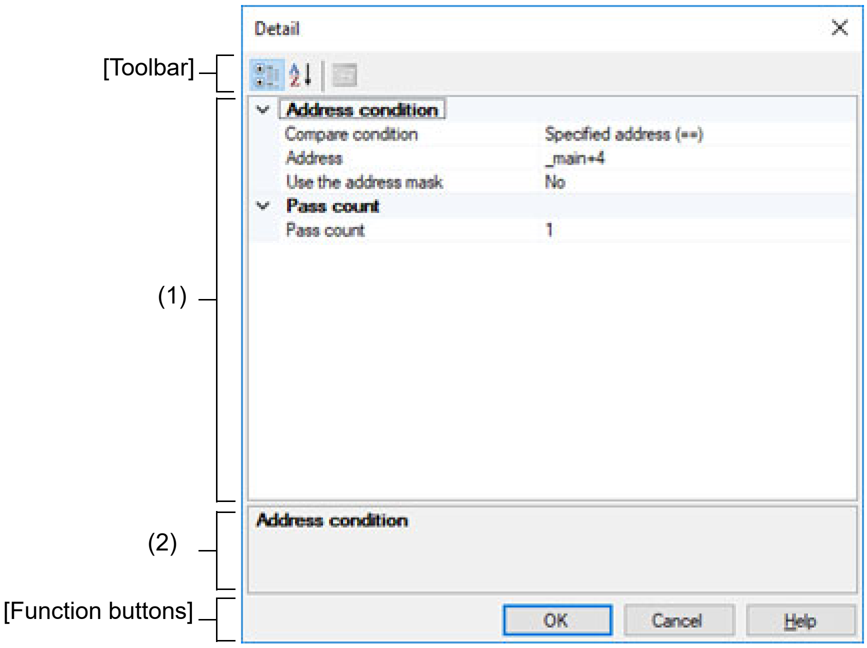


Figure A.43 Detail Dialog Box (for Execution Events) **[Simulator]**



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Function buttons]

[How to open]

- On the Events panel, move the caret to any one of the following events, and then select [Edit Condition...] from the context menu.
    - Hardware Break event (execution type)
    - Execution-related event as start and end condition in detailed information on Trace event
    - Execution-related event as start and end condition in detailed information on Timer Result event
    - Execution-related event as start and end condition in detailed information on Performance Measurement event

[Description of each area]

(1) Event conditions setting area

(a) [Address Condition]
Specify the address condition.

Compare condition	Specify the condition to compare address. [Full-spec emulator][E1][E20] The condition for an access event being generated varies with the value of this setting. [Specified address (==)] : Before Execution [Inside the range (<=Addresses<=)] / [Outside the range !(<=Addresses<=)] : After Execution		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	Specified address (==)	Specifies the address with [Address].
		Greater than or equal to (>=) [Simulator]	
		Less than or equal to (<=) [Simulator]	
		Inside the range (<=Addresses<=)	Specifies the range with [Start address] and [End address].
		Outside the range !(<=Addresses<=)	
Address	Specify the address. This item appears only when [Specified address (==)], [Greater than or equal to (>=)] or [Less than or equal to (<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Address expression within the valid range	
Start address	Specify the start address. This item appears only when [Inside the range (<=Addresses<=)] or [Outside the range !(<=Addresses<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Address expression within the valid range	

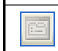
<table>
<tr><td rowspan="4">End address</td><td colspan="3">Specify the end address.
This item appears only when [Inside the range (<=Addresses<=)] or [Outside the range !(<=Addresses<=)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Address expression within the valid range</td></tr>
<tr><td rowspan="5">Use the address mask</td><td colspan="3">Specify whether to specify an address mask.
This item appears only when [Specified address (==)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Specifies an address mask.</td></tr>
<tr><td>No</td><td>Does not specify an address mask.</td></tr>
<tr><td rowspan="4">Mask value</td><td colspan="3">Specify the value of address mask.
This item appears only when [Use the address mask] is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Hexadecimal number up to five digits</td></tr>
</table>

(b) [Pass Count] **[Simulator]**
Specify the pass count condition.

<table>
<tr><td rowspan="4">Pass count</td><td colspan="2">Specify a pass count.
The relevant event occurs when the event condition is met as many times as the specified pass count.</td></tr>
<tr><td>Default</td><td>The current set value</td></tr>
<tr><td>Modifying</td><td>Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td>1 to 32768 in decimal number</td></tr>
</table>

(2) Description area
This area displays a simple description of the item selected in the Event conditions setting area.

[Toolbar]

	Displays all categories in the Event conditions setting area.
	Hides categories in the Event conditions setting area and rearranges only setting items in the ascending order
	This button is always invalid.

[Function buttons]

Button	Function
OK	Applies detailed settings made in this dialog box to execution-related events and closes this dialog box.
Cancel	Cancels the save and closes this dialog box.
Help	Displays the help for this dialog box.

Detail dialog box (for access events)

This dialog box is used to display and modify detailed information on an access-related event selected in the Events panel.

Figure A.44 Detail Dialog Box (for access Events) **[Full-spec emulator][E1][E20]**

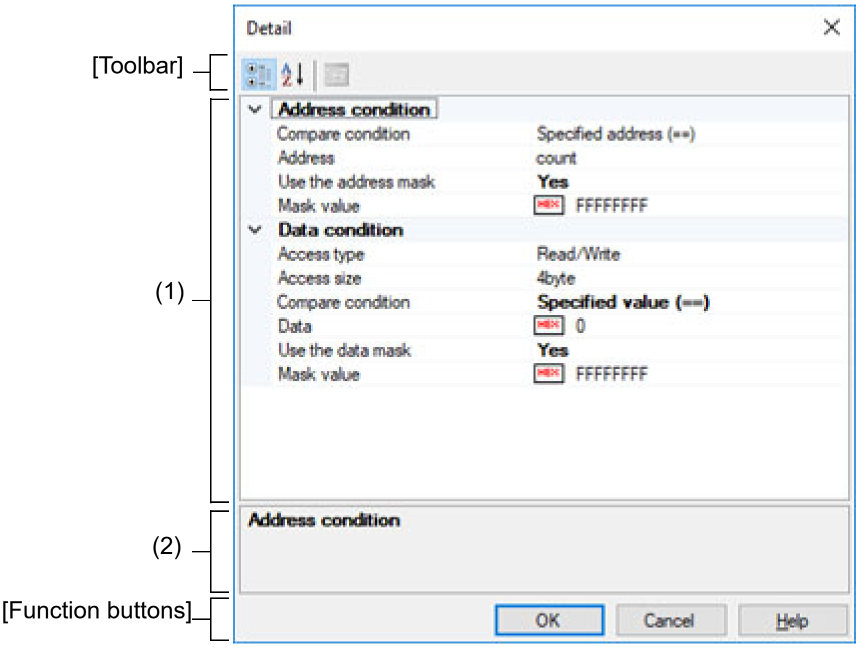
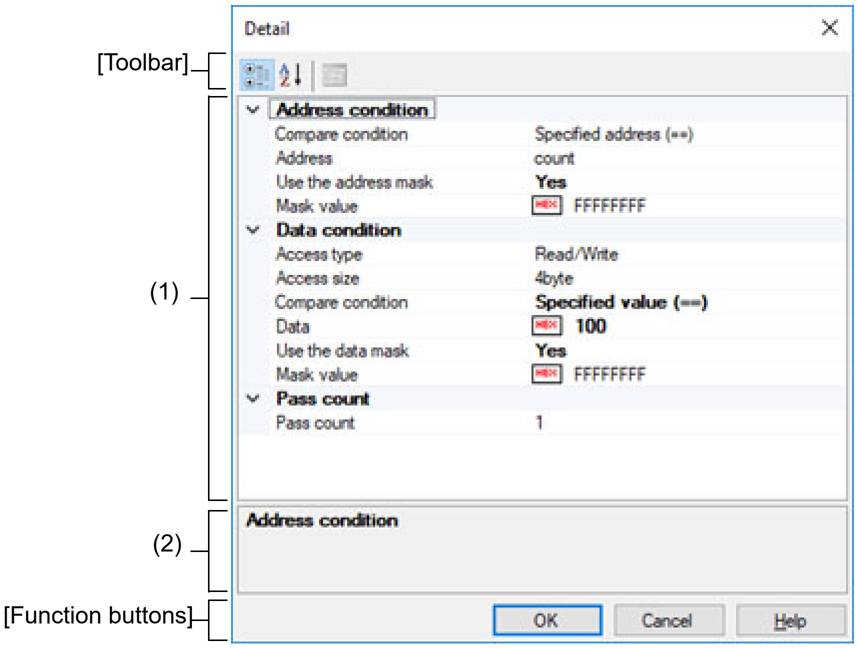


Figure A.45 Detail Dialog Box (for access Events) **[Simulator]**



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Function buttons]

[How to open]

- On the Events panel, move the caret to any one of the following events, and then select [Edit Condition...] from the context menu.
    - Hardware Break event (access type)
    - Access-related event as start and end condition in detailed information on Trace event
    - Access-related event as start and end condition in detailed information on Performance Measurement event

[Description of each area]

(1) Event conditions setting area

(a) [Address Condition]
Specify the address condition.

Compare condition	Specify the condition to compare address. [Full-spec emulator][E1][E20] The condition for an access event being generated varies with the value of this setting. [Specified address (==)] : Before Read, Before Write, Before Read/Write [Inside the range (<=Addresses<=)] / [Outside the range !(<=Addresses<=)] : After Read, After Write, After Read/Write		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	Specified address (==)	Specifies the address with [Address].
		Greater than or equal to (>=) [Simulator]	
		Less than or equal to (<=) [Simulator]	
		Inside the range (<=Addresses<=)	Specifies the range with [Start address] and [End address].
		Outside the range !(<=Addresses<=)	
Address	Specify the address. This item appears only when [Specified address (==)], [Greater than or equal to (>=)] or [Less than or equal to (<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Address expression within the valid range	
Start address	Specify the start address. This item appears only when [Inside the range (<=Addresses<=)] or [Outside the range !(<=Addresses<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Address expression within the valid range	

End address	Specify the end address. This item appears only when [Inside the range (<=Addresses<=)] or [Outside the range !(<=Addresses<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Address expression within the valid range	
Use the address mask	Specify whether to specify an address mask. This item appears only when [Specified address (==)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	Yes	Specifies an address mask.
		No	Does not specify an address mask.
Mask value	Specify the value of address mask. This item appears only when [Use the address mask] is set to [Yes].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to five digits	

(b) [Data Condition]
Specify the data condition.

Access type	Specify the type of access.		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	Read	Set a read access as the type of access.
		Write	Set a write access as the type of access.
		Read/Write	Sets a read and a write access as the type of access.

Access size	Specify the access size.		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	No conditions	Sets no access size. True for all access sizes. In [Compare condition] under [[Data Condition]], only [No conditions] is selectable.
		1byte	Sets 1-byte as the access size.
		2byte	Sets 2-byte as the access size.
		4byte	Sets 4-byte as the access size.
		8byte	[Full-spec emulator][E1][E20] Sets 8-byte as the access size.
		16byte	[Full-spec emulator][E1][E20] Sets 16-byte as the access size. In [Compare condition] under [[Data Condition]], only [No conditions] is selectable. This item only appears when the selected microcontroller supports it.
Compare condition	Specify the condition to compare the data. This item appears only when [1byte], [2byte], [4byte] or [8byte] is selected in [Access size].		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	No conditions	Specifies no data value.
		Specified value (==)	Specifies the data with [Data].
		Any other value (!=)	
		Greater than or equal to (>=) [Simulator]	
		Less than or equal to (<=) [Simulator]	
		Inside the range (<=Values<=) [Simulator]	Specifies the range with [Lower data] and [Upper data].
		Outside the range !(<=Values<=) [Simulator]	
Data	Specify the data to compare. This item appears only when [Specified value (==)], [Any other value (!=)], [Greater than or equal to (>=)] or [Less than or equal to (<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	

Lower data [Simulator]	Specify the lower data for the range in [Compare condition]. This item appears only when [Inside the range (<=Values<=)] or [Outside the range !(<=Values<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	
Upper data [Simulator]	Specify the upper data for the range in [Compare condition]. This item appears only when [Inside the range (<=Values<=)] or [Outside the range !(<=Values<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	
Use the data mask	Specify whether to specify a data mask. This item appears only when [1byte], [2byte], [4byte] or [8byte] is selected in [Access size]. This item appears only when [Specified value (==)] or [Any other value (!=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	Yes	Specifies an data mask.
		No	Does not specify an data mask.
Mask value	Specify the value of data mask. This item appears only when [Use the data mask] is set to [Yes].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	

(c) [Pass Count] **[Simulator]**

Specify the pass count condition.

Pass count	Specify a pass count. The relevant event occurs when the event condition is met as many times as the specified pass count.	
	Default	*The current set value*
	Modifying	Directly enter from the keyboard.
	Available values	1 to 32768 in decimal number

(2) Description area

This area displays a simple description of the item selected in the Event conditions setting area.

[Toolbar]

	Displays all categories in the Event conditions setting area.

	Hides categories in the Event conditions setting area and rearranges only setting items in the ascending order
	This button is always invalid.

[Function buttons]

Button	Function
OK	Applies detailed settings made in this dialog box to access-related events and closes this dialog box.
Cancel	Cancels the save and closes this dialog box.
Help	Displays the help for this dialog box.

Detailed Settings of Point Trace dialog box

This dialog box is used to display and change the detailed information on the point trace event selected on the Events panel. For details on point trace event setting, see "2.14 Collect Execution History of Programs".

Figure A.46 Detailed Settings of Point Trace Dialog Box **[Full-spec emulator][E1][E20]**

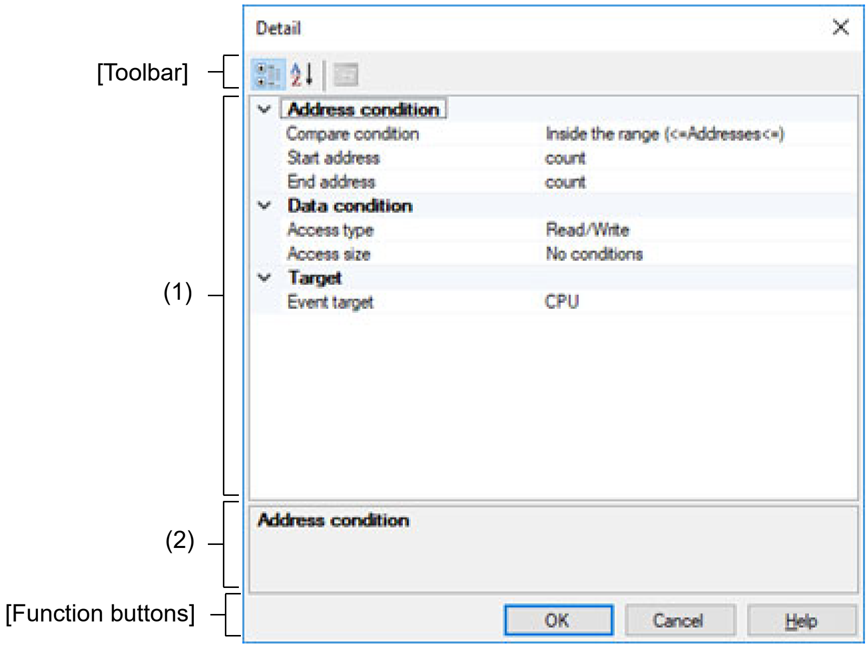
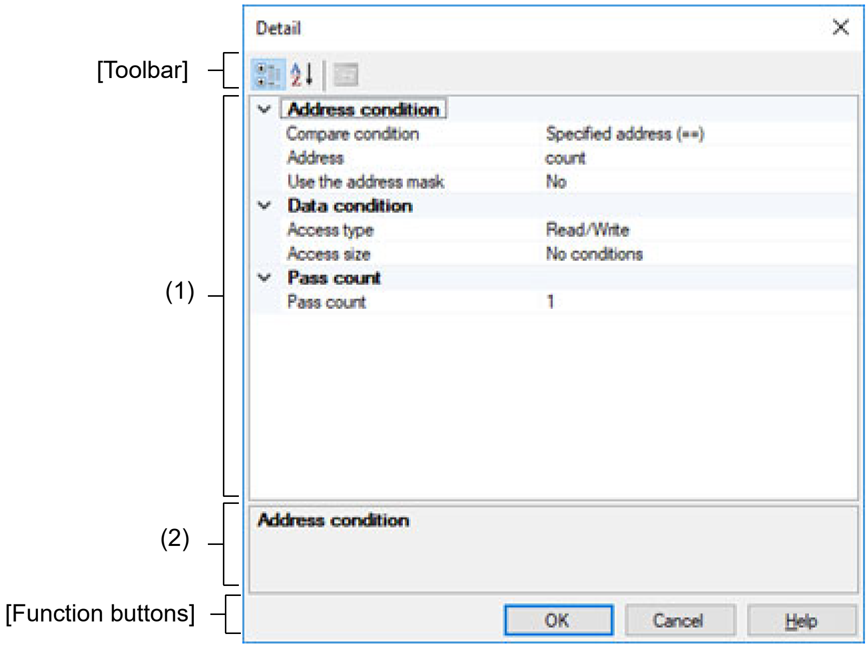


Figure A.47 Detailed Settings of Point Trace Dialog Box **[Simulator]**



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Description of each category]
- [Function buttons]

[How to open]

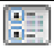
- On the Events panel, move the caret to the point trace event of which you wish to change the detailed information, then select [Edit Condition...] from the context menu.

[Description of each area]

(1) Detailed information display/change area
In this area, detailed information on the point trace 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) [Address condition]
You can display and modify the address condition of the point trace.

<table>
<tr><td rowspan="8">Compare condition</td><td colspan="3">Specify the condition to compare address.</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list</td></tr>
<tr><td rowspan="5">Available values</td><td>Specified address (==) [Simulator]</td><td rowspan="3">Specifies the address with [Address].</td></tr>
<tr><td>Greater than or equal to (>=) [Simulator]</td></tr>
<tr><td>Less than or equal to (<=) [Simulator]</td></tr>
<tr><td>Inside the range (<=Addresses<=)</td><td rowspan="2">Specifies the range with [Start address] and [End address].</td></tr>
<tr><td>Outside the range !(<=Addresses<=)</td></tr>
<tr><td rowspan="4">Address</td><td colspan="3">Specify the address.
This item appears only when [Specified address (==)], [Greater than or equal to (>=)] or [Less than or equal to (<=)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Address expression within the valid range</td></tr>
</table>

<table>
<tr><td rowspan="4">Start address</td><td colspan="3">Specify the start address.
This item appears only when [Inside the range (<=Addresses<=)] or [Outside the range !(<=Addresses<=)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Address expression within the valid range</td></tr>
<tr><td rowspan="4">End address</td><td colspan="3">Specify the end address.
This item appears only when [Inside the range (<=Addresses<=)] or [Outside the range !(<=Addresses<=)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Address expression within the valid range</td></tr>
<tr><td rowspan="5">Use the address mask</td><td colspan="3">Specify whether to specify an address mask.
This item appears only when [Specified address (==)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list</td></tr>
<tr><td rowspan="2">Available values</td><td>Yes</td><td>Specifies an address mask.</td></tr>
<tr><td>No</td><td>Does not specify an address mask.</td></tr>
<tr><td rowspan="4">Mask value</td><td colspan="3">Specify the value of address mask.
This item appears only when [Use the address mask] is set to [Yes].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Hexadecimal number up to five digits</td></tr>
</table>

(2) [Data condition]
You can display and modify the data condition of the point trace.

<table>
<tr><td rowspan="6">Access type</td><td colspan="3">Specify the type of access.</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list</td></tr>
<tr><td rowspan="3">Available values</td><td>Read</td><td>Set a read access as the type of access.</td></tr>
<tr><td>Write</td><td>Set a write access as the type of access.</td></tr>
<tr><td>Read/Write</td><td>Sets a read and a write access as the type of access.</td></tr>
</table>

<table>
<tr><td rowspan="9">Access size</td><td colspan="3">Specify the access size.</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list</td></tr>
<tr><td rowspan="6">Available values</td><td>No conditions</td><td>Sets no access size.
True for all access sizes.
In [Compare condition] under [[Data condition]], only [No conditions] is selectable.</td></tr>
<tr><td>1byte</td><td>Sets 1-byte as the access size.</td></tr>
<tr><td>2byte</td><td>Sets 2-byte as the access size.</td></tr>
<tr><td>4byte</td><td>Sets 4-byte as the access size.</td></tr>
<tr><td>8byte</td><td>[Full-spec emulator][E1][E20]
Sets 8-byte as the access size.</td></tr>
<tr><td>16byte</td><td>[Full-spec emulator][E1][E20]
Sets 16-byte as the access size.
In [Compare condition] under [[Data condition]], only [No conditions] is selectable.
This item only appears when the selected microcontroller supports it.</td></tr>
<tr><td rowspan="10">Compare condition</td><td colspan="3">Specify the condition to compare the data.
This item appears only when [1byte], [2byte], [4byte] or [8byte] is selected in [Access size].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list</td></tr>
<tr><td rowspan="7">Available values</td><td>No conditions</td><td>Specifies no data value.</td></tr>
<tr><td>Specified value (==)</td><td rowspan="4">Specifies the data with [Data].</td></tr>
<tr><td>Any other value (!=)</td></tr>
<tr><td>Greater than or equal to (>=)
[Simulator]</td></tr>
<tr><td>Less than or equal to (<=)
[Simulator]</td></tr>
<tr><td>Inside the range (<=Values<=)
[Simulator]</td><td rowspan="2">Specifies the range with [Lower data] and [Upper data].</td></tr>
<tr><td>Outside the range !(<=Values<=)
[Simulator]</td></tr>
<tr><td rowspan="4">Data</td><td colspan="3">Specify the data to compare.
This item appears only when [Specified value (==)], [Any other value (!=)], [Greater than or equal to (>=)] or [Less than or equal to (<=)] is selected in [Compare condition].</td></tr>
<tr><td>Default</td><td colspan="2">The current set value</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td>Available values</td><td colspan="2">Hexadecimal number up to [Access size]</td></tr>
</table>

Lower data [Simulator]	Specify the lower data for the range in [Compare condition]. This item appears only when [Inside the range (<=Values<=)] or [Outside the range !(<=Values<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	
Upper data [Simulator]	Specify the upper data for the range in [Compare condition]. This item appears only when [Inside the range (<=Values<=)] or [Outside the range !(<=Values<=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	
Use the data mask	Specify whether to specify a data mask. This item appears only when [1byte], [2byte], [4byte] or [8byte] is selected in [Access size]. This item appears only when [Specified value (==)] or [Any other value (!=)] is selected in [Compare condition].		
	Default	*The current set value*	
	Modifying	Select from the drop-down list	
	Available values	Yes	Specifies an data mask.
		No	Does not specify an data mask.
Mask value	Specify the value of data mask. This item appears only when [Use the data mask] is set to [Yes].		
	Default	*The current set value*	
	Modifying	Directly enter from the keyboard.	
	Available values	Hexadecimal number up to [Access size]	

(3) [Pass Count] **[Simulator]**
Specify the pass count condition.

Pass count	Specify a pass count. The relevant event occurs when the event condition is met as many times as the specified pass count.	
	Default	*The current set value*
	Modifying	Directly enter from the keyboard.
	Available values	1 to 32768 in decimal number

(4) [Target] **[Full-spec emulator][E1][E20]**
You can display and modify the target of the point trace.

Event target	Specify the target of the point trace.	
	Default	*The current set value*
	Modifying	Select from the drop-down list.
	Available values	One of the following as selected from the drop-down list. CPU, Global RAM, Cluster RAM
Cluster No.	Specify the cluster number which is the target of the point trace.	
	Default	*The current set value*
	Modifying	Select from the drop-down list.
	Available values	*Depends on the device*

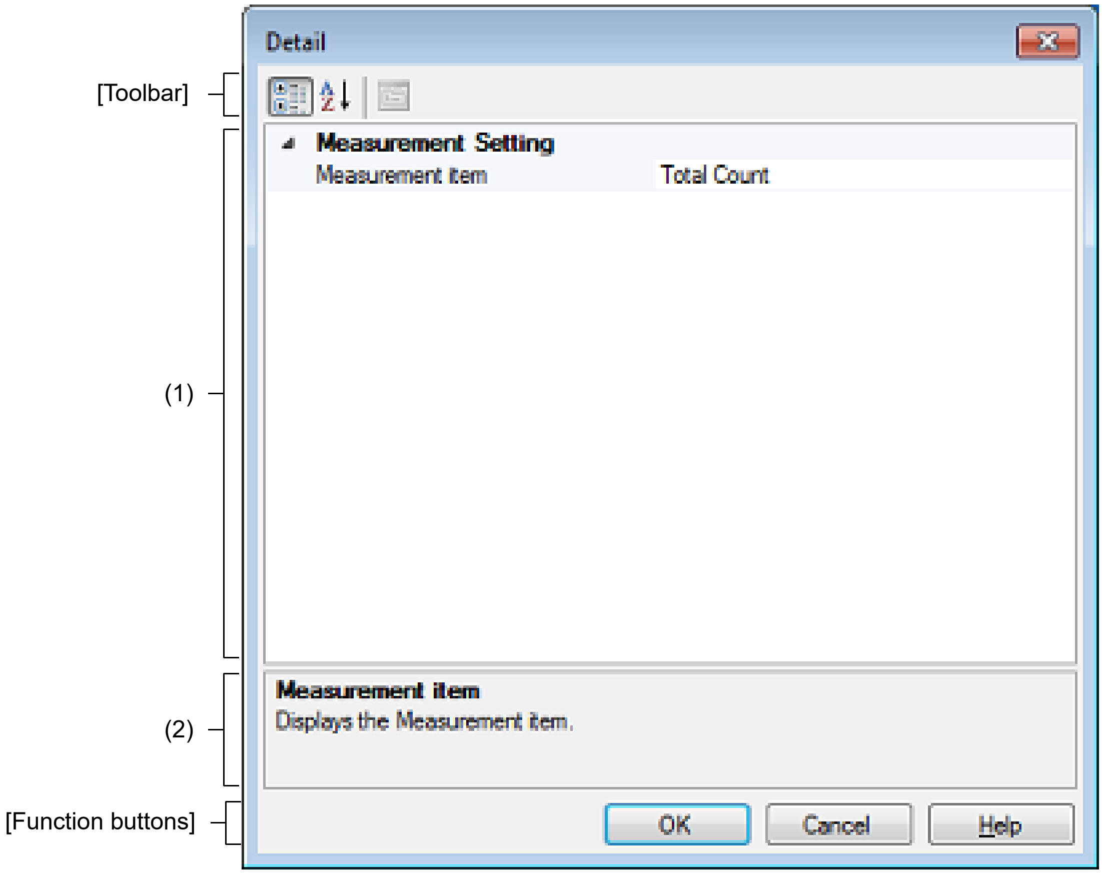
[Function buttons]

Button	Function
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 Timer Measurement dialog box [Full-spec emulator][E1][E20]

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 "2.15 Measure Execution Time of Programs".

Figure A.48 Detailed Settings of Timer Measurement Dialog Box [Full-spec emulator][E1][E20]



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Description of each category]
- [Function buttons]

[How to open]

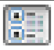
- On the Events panel, move the caret to the timer event of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.

[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	Specify the measurement item.	
	Default	*The current set value*
	Modifying	Select from the drop-down list.
	Available values	One of the following as selected from the drop-down list. Total Count, Max Count, Min Count, Pass Count

[Function buttons]

Button	Function
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 Performance Measurement dialog box [Full-spec emulator][E1][E20]

This dialog box is used to display and change the detailed information on the Performance Measurement event selected on the Events panel. Note that you cannot edit the address value of the Performance Measurement event in this dialog box. If you need to edit the address value, first delete the Performance Measurement event and then create a new one. For details on Performance Measurement event setting, see "2.16 Measure Performance [Full-spec emulator][E1][E20]".

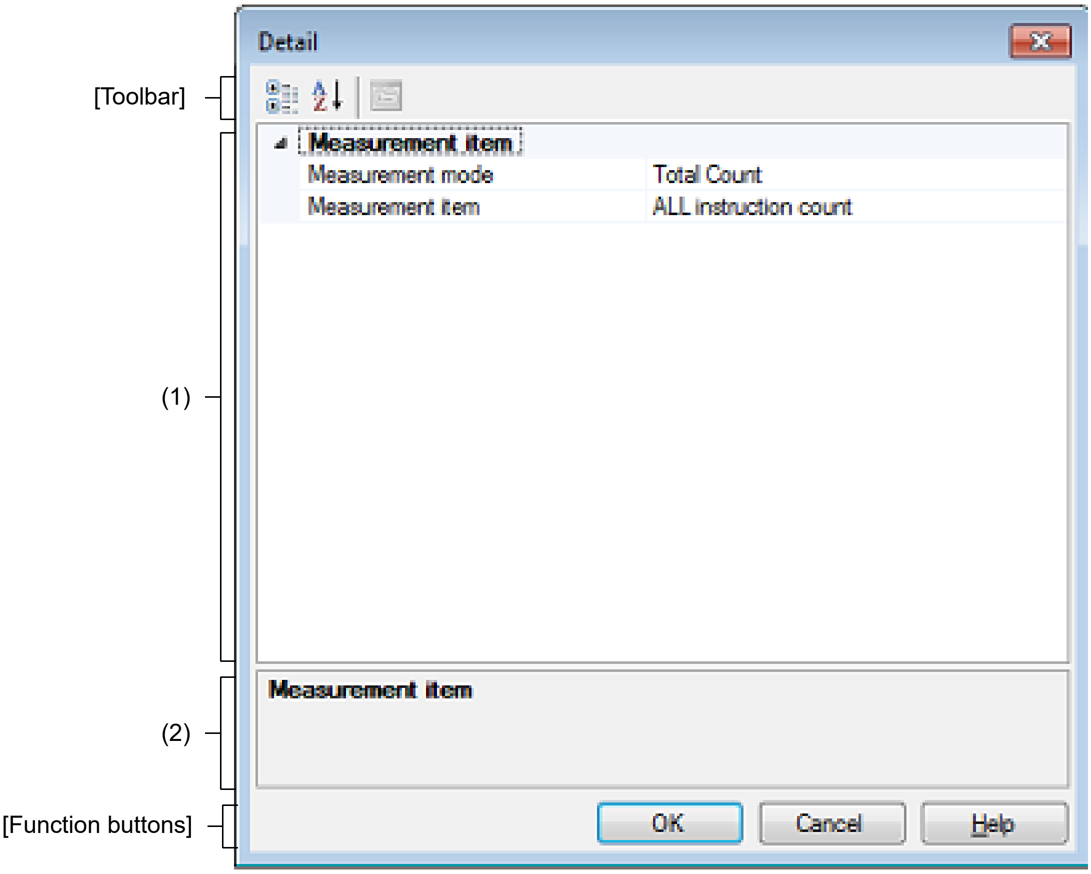


Figure A.49 Detailed Settings of Performance Measurement Dialog Box [Full-spec emulator][E1][E20]

This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]
- [Description of each category]
- [Function buttons]

[How to open]

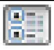
- On the Events panel, move the caret to the Performance Measurement event of which you wish to change the detailed information, then select [Edit Condition ...] from the context menu.

[Description of each area]

(1) Detailed information display/change area
In this area, detailed information on the Performance Measurement 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 performance measurement.

Measurement mode	Specify the measurement mode.	
	Default	*The current set value*
	Modifying	Select from the drop-down list.
	Available values	One of the following as selected from the drop-down list. - Total Count - Max Count - Min Count - New Count - Pass Count

Measurement item	Specify the measurement item.	
	Default	*The current set value*
	Modifying	Select from the drop-down list.
	Available values	One of the following as selected from the drop-down list. [RH850G3M, RH850G3K, RH850G3MH, RH850G3KH] - ALL instruction count - Branch instruction count - EI level interrupt count - FE level interrupt count - ALL instruction async exception count - ALL instruction sync exception count - Clock cycle - Non-interrupt cycle - Interrupt disable cycle of DI/EI - CPU issued instruction fetch request count - Response count for CPU issued instruction fetch request - Flash ROM data request count [RH850G4MH, RH850G4KH] - ALL instruction count - Branch instruction count (excluding condition mismatch Bcond instruction, Loop instruction, exception instruction) - Conditional branch instruction count (Bcond instruction, Loop instruction) - Branch prediction misses of conditional branch instruction count (Bcond instruction, Loop instruction) - EI level interrupt count - FE level interrupt count - ALL instruction async exception count - ALL instruction sync exception count - Stall cycles issued to the instruction execution unit - Clock cycle - Non-interrupt cycle - Interrupt disable cycle of DI/EI - CPU issued instruction fetch request count - Response count for CPU issued instruction fetch request

Caution When “Pass Count” is selected in [Measurement mode], [Measurement item] is not displayed.

[Function buttons]

Button	Function
OK	Applies the detailed settings specified in the dialog box to the Performance Measurement event and closes this dialog box.

Button	Function
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

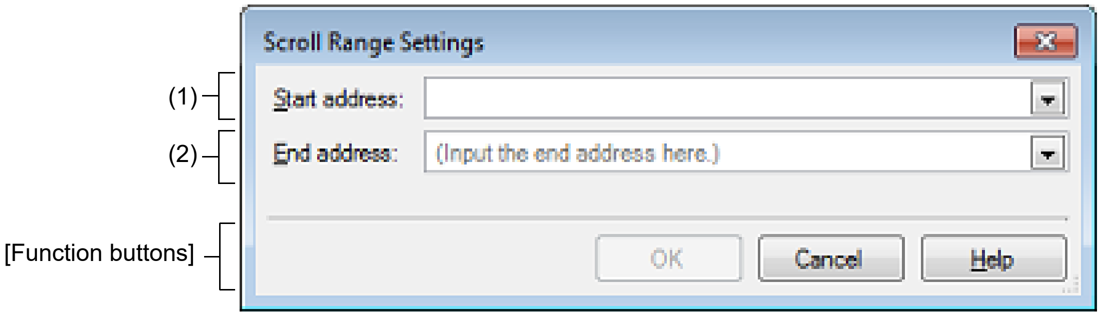
Scroll Range Settings dialog box

This dialog box is used to set the scroll range of the vertical scroll bar on the Memory panel/Disassemble panel.
By setting the appropriate range, it is possible to improve the operability of a mouse (e.g. dragging) because the size of the vertical scroll bar on the panel is changed suitably.

Caution After setting a scroll range via this dialog box, the scroll range is not updated automatically even if the address evaluated by the address expression is changed because of such as a line assembly.

Remark It is possible to move outside the scroll range by using the [Page Up]/[Page Down]/[Up]/[Down] key, a button at either end of the scroll bar or a menu item related to the jump function.

Figure A.50 Scroll Range Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Memory panel, click the button from [View] on the toolbar.
- On the Memory panel, select the [View] menu >> [Settings Scroll Range...] from the context menu.
- On the Disassemble panel, click the button from [View] on the toolbar.
- On the Disassemble panel, select the [View] menu >> [Settings Scroll Range...] from the context menu.

[Description of each area]

(1) [Start address] area
Specify the start address of the range of scrolling.
You can either type an address expression directly into the text box (up to 1024 characters), or select it from the input history via the drop-down list (up to 10 items).
Note that the setting of the scroll range is not performed if "All" is selected in the drop-down list at this time (the scroll range is not limited).

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (see "2.21.2 Symbol name completion function").

(2) [End address] area
Specify the end address of the range of scrolling.
You can either type an address expression directly into the text box (up to 1024 characters), or select it from the input history via the drop-down list (up to 10 items).
Note that this area becomes invalid if [Start address] is specified with [All].

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (see "2.21.2 Symbol name completion function").

[Function buttons]

Button	Function
OK	Sets the specified scroll range for the target panel. Moves the caret to the start address, from the beginning of the area displayed in the target panel.
Cancel	Cancels the jump and closes this dialog box.
Help	Displays the help for this dialog box.

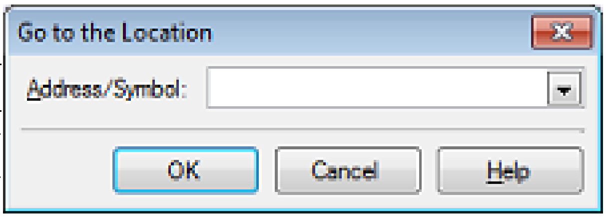
Go to the Location dialog box

This dialog box is used to move the caret to a specified position.

Figure A.51 Go to the Location Dialog Box

(1)

[Function buttons]



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Focus the Disassemble panel, and then select [Go to…] from the [Edit] menu.
- Focus the IOR panel, and then select [Go to…] from the [Edit] menu.
- On the Disassemble panel, select [Go to…] from the context menu.
- On the IOR panel, select [Go to…] from the context menu.

[Description of each area]

(1) [Address/Symbol], or [IOR] area

Specify the location to which the caret jumps.
You can either type a location directly into the text box (up to 1024 characters), or select one from the input history via the drop-down list (up to 10 items).
The data to specify various depending on the target panel, as follows:

Target Panel	Data Specified
Disassemble panel	Address expression
IOR panel	I/O register name

Remark If this dialog box is opened from the Disassemble panel, a symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in this text box (see "2.21.2 Symbol name completion function").

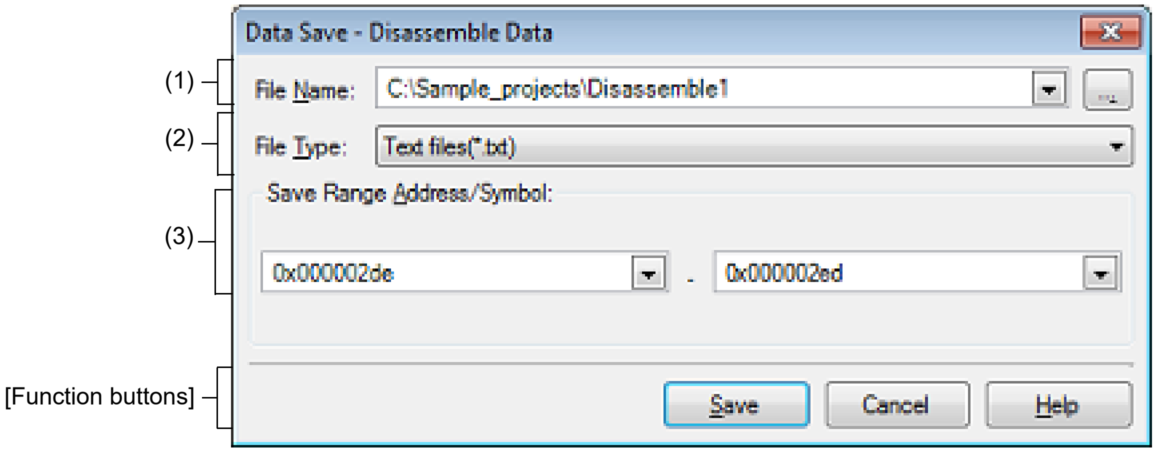
[Function buttons]

Button	Function
OK	Moves the caret to the specified location, from the beginning of the area displayed in the target panel.
Cancel	Cancels the jump and closes this dialog box.
Help	Displays the help for this dialog box.

Data Save dialog box

This dialog box is used to save data displayed in the Disassemble panel, Memory panel, or Trace panel, and save uploaded data (see "2.6.3 Execute uploading").
This dialog box appears only when connected to the debug tool.

Figure A.52 Data Save Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- With the Disassemble panel in focus, select [Save Disassemble Data As…] form the [File] menu.
- With the Memory panel in focus, select [Save Memory Data As…] form the [File] menu.
- With the Trace panel in focus, select [Save Trace Data As…] form the [File] menu.
- From the [Debug] menu, select [Upload…].

[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 […] button, and selecting a file via the Select 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) [File Type] area
Select the format in which to save the file from the following drop-down list.
The available file formats will differ as follows depending on the type of data being saved.

(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

Intel HEX format (*.hex)	Intel HEX file format
Motorola S-record (*.mot)	Motorola S-record file format
Binary data (*.bin)	Binary file format

Remark See "2.6.3 Execute uploading" for details on uploading.

(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 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.
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 A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] key in each text box (see "2.21.2 Symbol name completion function").

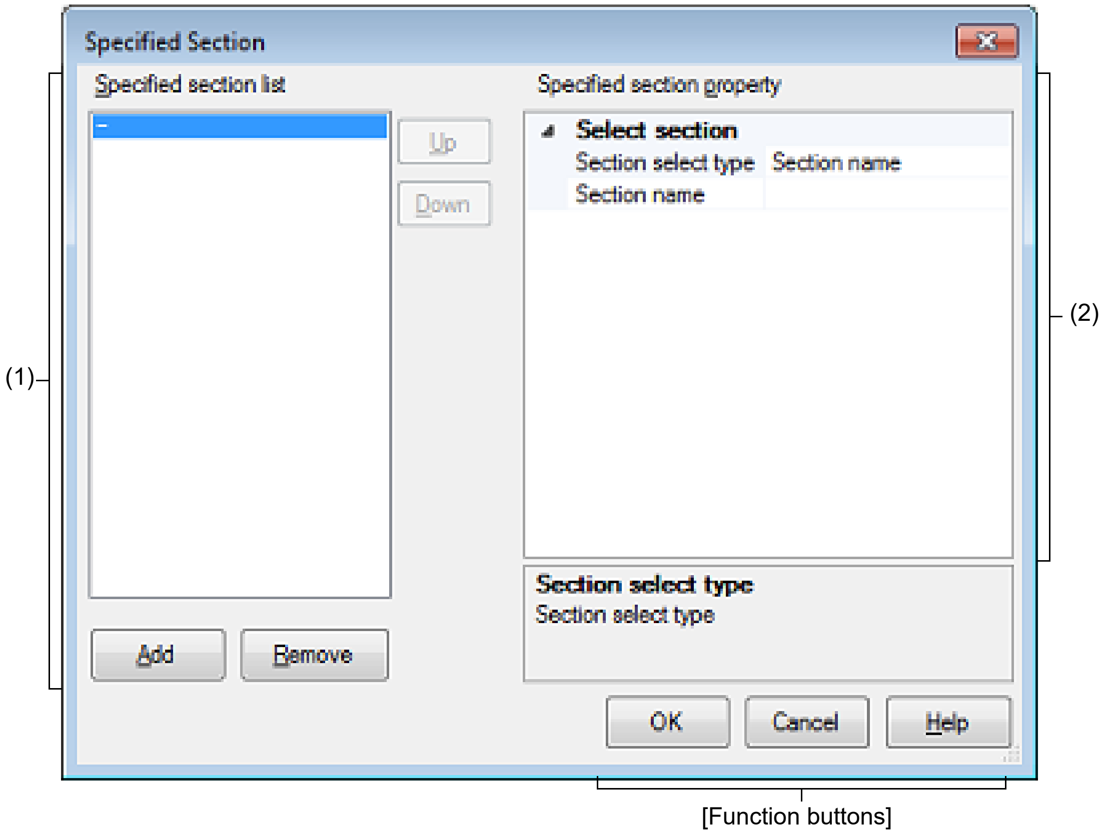
[Function buttons]

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.

Specified Section dialog box

This dialog box is used to specify the range for skipping step execution.

Figure A.53 Specified Section Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the [Debug Tool Settings] tab of the Property panel, click the [...] button displayed by selecting the [Target section] property in the [Step function] category.

[Description of each area]

(1) [Specified section list] area

(a) List display
This area is used to display the list of the range for skipping step execution.
To add a new section, click the [Add] button in this area, then in the [Specified section property] area, specify the sections to add.

(b) Button

Button	Function
Up	Moves the selected section up one row in the list. Clicking this for the top section in the list has no effect.

Button	Function
Down	Moves the selected section down one row in the list. Clicking this for the bottom section in the list has no effect.
Add	Adds an empty item "-" to the list, and selects it. Specify the section to add in the [Specified section property] area.
Remove	Removes the selected section from the list.

(2) [Specified section property] area

(a) [Select section]

This area is used to display or edit the section selected in the [Specified section list] area.
It can also be used to specify the new sections added via the [Add] button.

<table>
<tr><td rowspan="5">Section select type</td><td colspan="3">Specify the method for specifying the section.</td></tr>
<tr><td>Default</td><td colspan="2">Section name</td></tr>
<tr><td>Modifying</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Available values</td><td>Section name</td><td>Specifies the range by the section name.</td></tr>
<tr><td>Start and end address</td><td>Specifies the range by the start and end address.</td></tr>
<tr><td rowspan="2">Section name</td><td colspan="3">Specify the section name.
Note that this item appears only when [Section select type] is set to [Section name].</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td rowspan="2">Start address</td><td colspan="3">Specify the start address.
Note that this item appears only when [Start and end address] is set to [Section name].</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
<tr><td rowspan="2">End address</td><td colspan="3">Specify the end address.
Note that this item appears only when [Start and end address] is set to [Section name].</td></tr>
<tr><td>Modifying</td><td colspan="2">Directly enter from the keyboard.</td></tr>
</table>

[Function buttons]

Button	Function
OK	Finishes configuring the target sections, and closes this dialog box.
Cancel	Cancels any changes to the target sections, 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.

When the microcontroller used is a single-core product of RH850, start the exclusive control checking tool to check whether exclusive control for variables is performed correctly. This tool can also be used to confirm whether variables are accessed correctly in the exclusive control period.
See "2.23 Exclusive Control Checking Tool" for detail about the exclusive control checking tool.

Caution The exclusive control checking tool can be used when CC-RH V1.04.00 or later is in use and Full-spec emulator, E1 or E20 is selected as the debug tool.

Figure A.54 Functions and Variables Access Table Panel: [Orthogonal Table] Tab

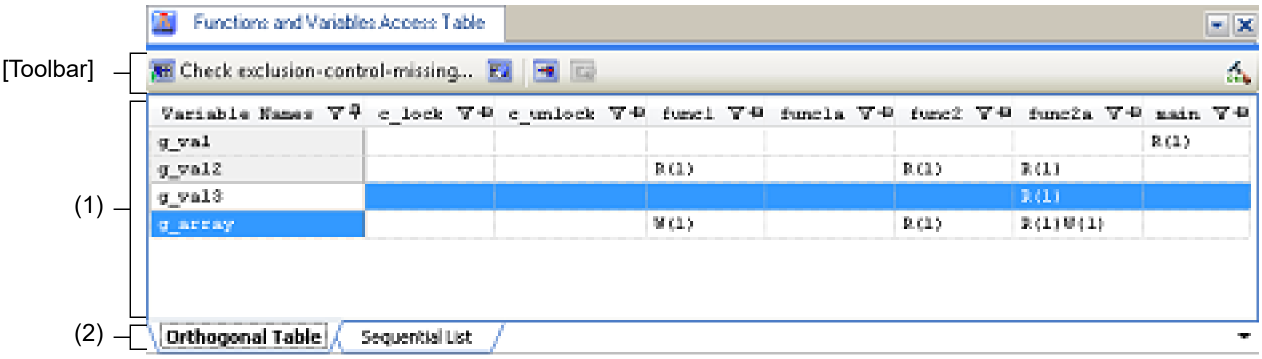
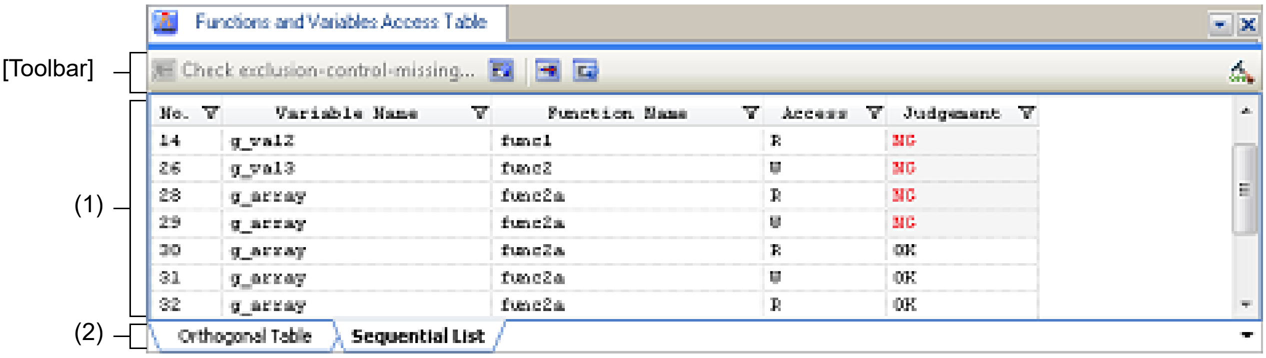


Figure A.55 Functions and Variables Access Table Panel: [Sequential List] Tab



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].
- On the Solution List panel, click the [Go] button of Exclusive Control Checking Tool.

[Description of each area]

(1) Table area
The contents of this area are switched according to the selected tab.

(a) When the [Orthogonal Table] tab is selected
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.
- A cell whose letters are in error color indicates a location at which there is a problem in exclusive control.

Double-clicking the variable name or cell which shows the state of access to the variable displays that definition in an editor.

(b) When the [Sequential List] tab is selected
The information obtained by executing the exclusive control checking tool (which variable is accessed by the function, whether that access is read or write, and whether access was made correctly during the exclusive control section) is displayed in time series.

- 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.
- A cell whose letters are in error color indicates a location at which there is a problem in exclusive control.

Double-clicking the variable name, function name, or cell which shows the state of access to the variable displays that definition in an editor.

(2) Tab selection area
There are two tabs: [Orthogonal Table] tab which displays the functions that access variables in the form of an orthogonal table and [Sequential List] tab which displays time-series information on exclusive control that was measured by the exclusive control checking tool.
The [Sequential List] tab only appears when the target microcontroller is a single-core product of RH850 and used with an emulator.

[Toolbar]

The toolbar is displayed only when the target microcontroller is a single-core product of RH850 and used with an emulator, and the compiler in use supports generation of cross reference information.

	Displays the Exclusive Control Checking Tool dialog box used to perform checking of exclusive control for variables selected in the [Orthogonal Table] tab. This button is valid only when all of the following conditions are met. - Building is not being performed. - The debug tool is not running. - The [Orthogonal Table] tab is selected. - At least one line is selected.
	Clears the result detected by the exclusive control checking tool.
	Displays only variables for which exclusive control did not work was found by the checking tool.
	When this button is turned ON, continuous lines with the same contents are merged into one line on the [Sequential List] tab. This is valid only when the [Sequential List] tab is selected.

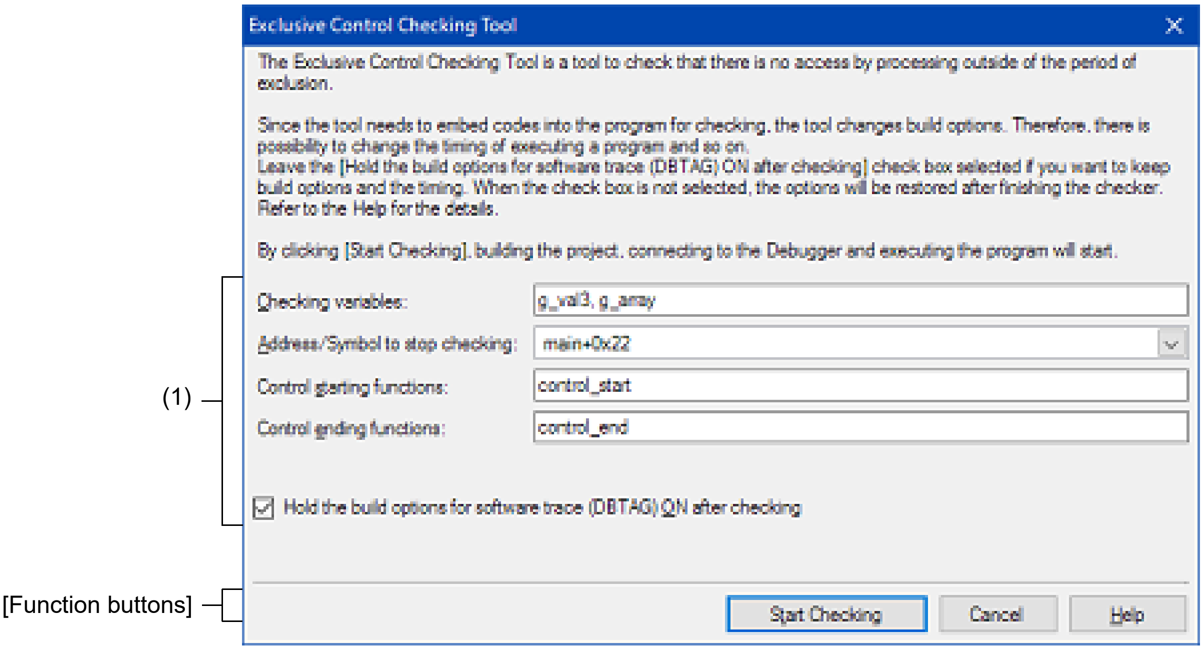
	If the build option related to generation of the cross reference information is invalid, clicking on this button validates the build option and runs a build. If a file is being edited by the text editor at the time, the file is saved.

Exclusive Control Checking Tool dialog box

This dialog box is used to check whether exclusive control is correctly performed for a certain global variable (except static variable) and make the settings required for checking. The checking result is displayed in the Functions and Variables Access Table panel.
See "2.23 Exclusive Control Checking Tool" for detail.

Caution This tool can be supported when CC-RH V1.04.00 or later is in use.

Figure A.56 Exclusive Control Checking Tool Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Functions and Variables Access Table panel, click the button on the toolbar.

[Description of each area]

(1) Setting area
Make settings for checking of exclusive control.

(a) [Checking variables]
Specify variable names which are targets of checking whether exclusive control has been performed correctly. The variables selected on the Functions and Variables Access Table panel are specified by default. When specifying multiple variables, separate them by commas.

(b) [Address/Symbol to stop checking]
Checking of exclusive control is carried out by executing the user program. Specify an address or function symbol where checking will end.
You can either type a hexadecimal number or address expression directly into the text box, or select one from the input history via the drop-down list (up to 10 items).

Remark A symbol name at the current caret position can be complemented by pressing the [Ctrl] + [Space] keys in this text box.

(c) [Control starting functions]/[Control ending functions]
Specify the function for starting exclusive control for the selected variables in [Control starting functions].
Specify the function for ending exclusive control in [Control ending functions].
When specifying multiple functions, separate them by commas.

(d) [Hold the build options for software trace (DBTAG) ON after checking]
The exclusive control checking tool embeds instructions in the program to check whether exclusive control is performed correctly. If operation after checking has ended needs to be the same as that during checking, select this check box to leave the instructions embedded in the program.

[Function buttons]

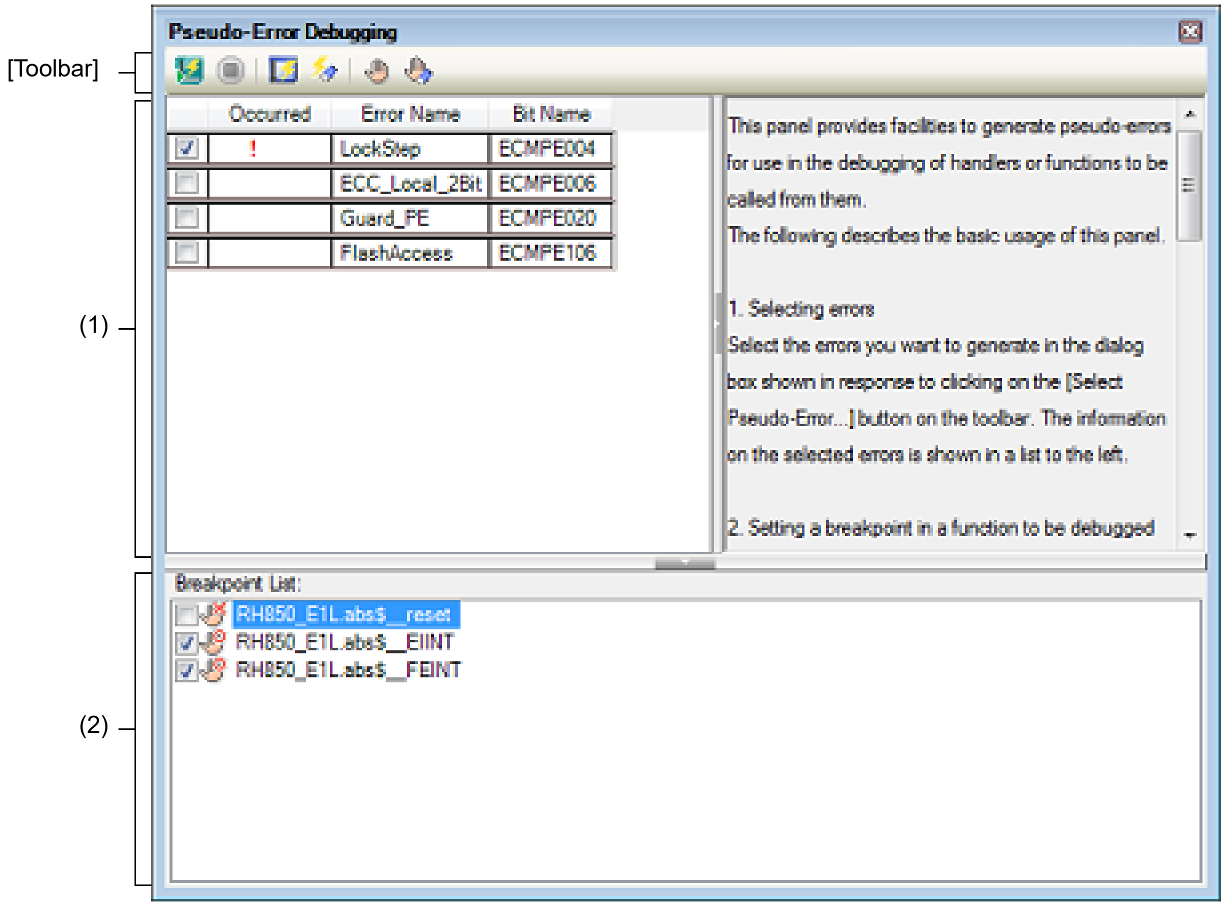
Button	Function
Start Checking	Starts checking of exclusive control for the variables specified in [Checking variables]. After checking has ended, the result of whether exclusive control is performed correctly is reflected to the Functions and Variables Access Table panel is closed. If a file is being edited by the text editor when checking is started, the file is saved.
Cancel	Closes this dialog box without performing checking of exclusive control.
Help	Displays the help for this dialog box.

Pseudo-Error Debugging panel [Full-spec emulator][E1][E20]

This panel is central to the functionality of the solution for pseudo-error debugging.

See "2.24 Pseudo-error Debugging [Full-spec emulator][E1][E20]" for details on the solution for pseudo-error debugging.

Figure A.57 Pseudo-Error Debugging Panel



This section describes the following.

- [How to open]
- [Description of each area]
- [Toolbar]

[How to open]

- From the [Debug] menu, select [Debug Solutions] >> [Pseudo-Error Debugging].
- On the Solution List panel, click the [Pseudo-error Debugging] button of Pseudo-error debugging.

[Description of each area]

(1) Pseudo-error list area
This area lists pseudo-errors which are targets in pseudo-error debugging.
The full name of a pseudo-error is displayed by hovering the mouse cursor over the node of each pseudo-error.
Each item is explained as follows:

(a) Check box
When a check box is selected, this pseudo-error is generated after pseudo-error debugging starts.

(b) Occurred
This column shows whether this pseudo-error has been generated when the debugger is stopped.
"!" is displayed if the pseudo-error has been generated and nothing is displayed if it has not been generated.

Remark A bit is always assigned per error to indicate whether an error occurred. Two bits are assigned to some errors and the number of occurrences of an error that can be counted is up to three (e.g., this is the case for errors No. 0 to 7 in RH850/G4MH devices). If the pseudo-error has occurred several times, “!” will be displayed as the number of times the error has occurred. To check the number of bits that are assigned to an error, see “ECM Error Source Status Register” in the chapter of the ECM in User's Manual: Hardware for the MCU in use.

(c) Error Name
The abbreviated name of this pseudo-error is displayed.

(d) Bit Name
The bit name of IOR that triggers the generation of this pseudo-error is displayed.

(2) [Breakpoint List]
A list of breakpoints that are to be used for pseudo-error debugging is displayed.
Only the breakpoints whose check box is selected will be valid for pseudo-error debugging.
Either of the following icons is displayed at the beginning of each breakpoint.

	Breakpoint is valid.
	Breakpoint is invalid.

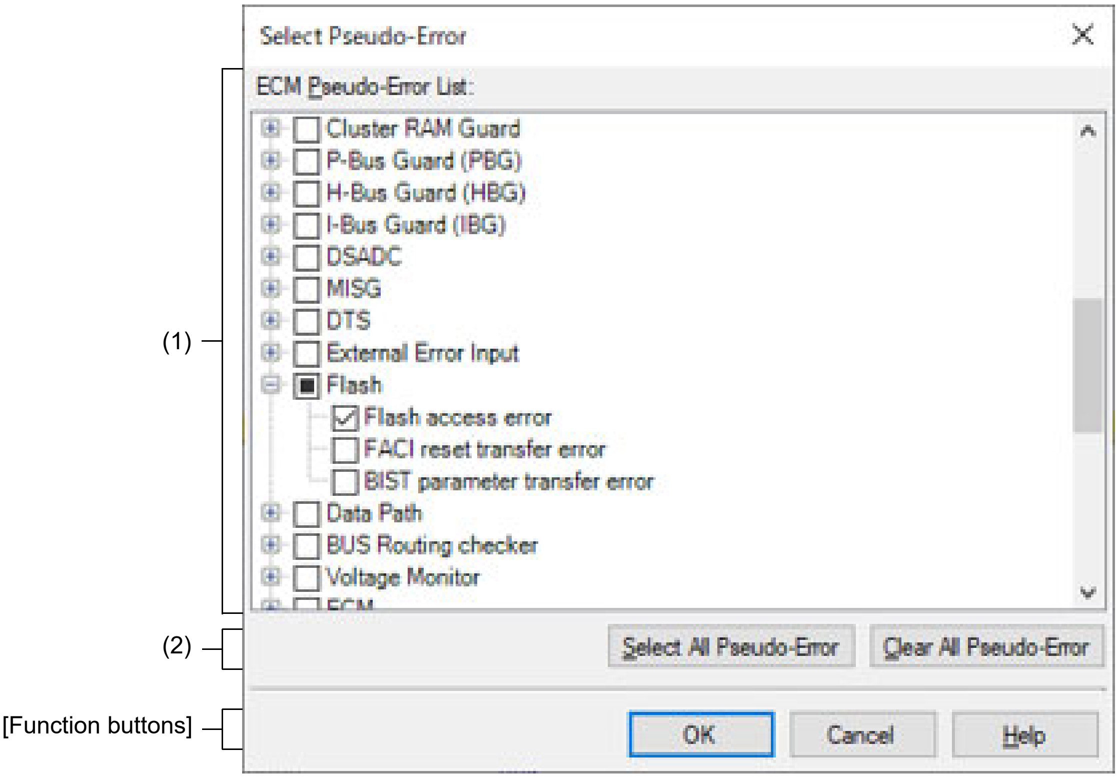
[Toolbar]

	Starts pseudo-error debugging based on the settings made in this panel.
	Has the same functionality as the same button on the Debug toolbar in the Main window.
	Opens the Select Pseudo-Error dialog box [Full-spec emulator][E1][E20] used to select ECM pseudo-errors that are targets of pseudo-error debugging.
	Clears the error status of all pseudo-errors.
	Opens the Breakpoint Setting dialog box [Full-spec emulator][E1][E20] to add a breakpoint.
	Deletes breakpoints selected in [Breakpoint List].

Select Pseudo-Error dialog box [Full-spec emulator][E1][E20]

This dialog box is used to select ECM pseudo-errors displayed in the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].

Figure A.58 Select Pseudo-Error Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20], click the [Select Pseudo-Error...] button on the toolbar.

[Description of each area]

(1) [ECM Pseudo-error List]
A list of pseudo-errors managed by the ECM is displayed.
When a check box is selected, the pseudo-error is regarded as a target in the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].

(2) Button area

Select All Pseudo-Error	Selects check boxes of all pseudo-errors in [ECM Pseudo-Error List].
Clear All Pseudo-Error	Clears check boxes of all pseudo-errors in [ECM Pseudo-Error List].

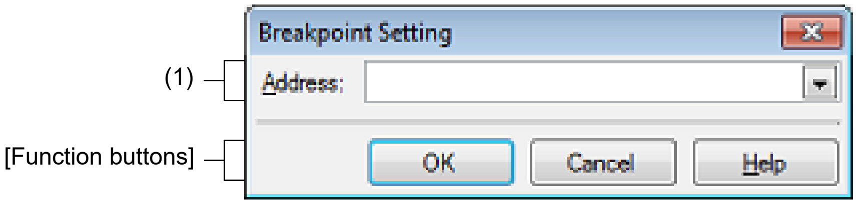
[Function buttons]

Button	Function
OK	Registers the pseudo-errors that were selected in this dialog box to the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20] and deletes the unselected pseudo-errors from the panel.
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Breakpoint Setting dialog box [Full-spec emulator][E1][E20]

This dialog box is used to set the breakpoints that are to be registered to the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].

Figure A.59 Breakpoint Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20], click the [Set Breakpoint...] button on the toolbar.

[Description of each area]

(1) [Address]
You can either enter an address expression of a breakpoint directly into the text box (up to 1024 characters), or select one from the input history via the drop-down list (up to 10 items).

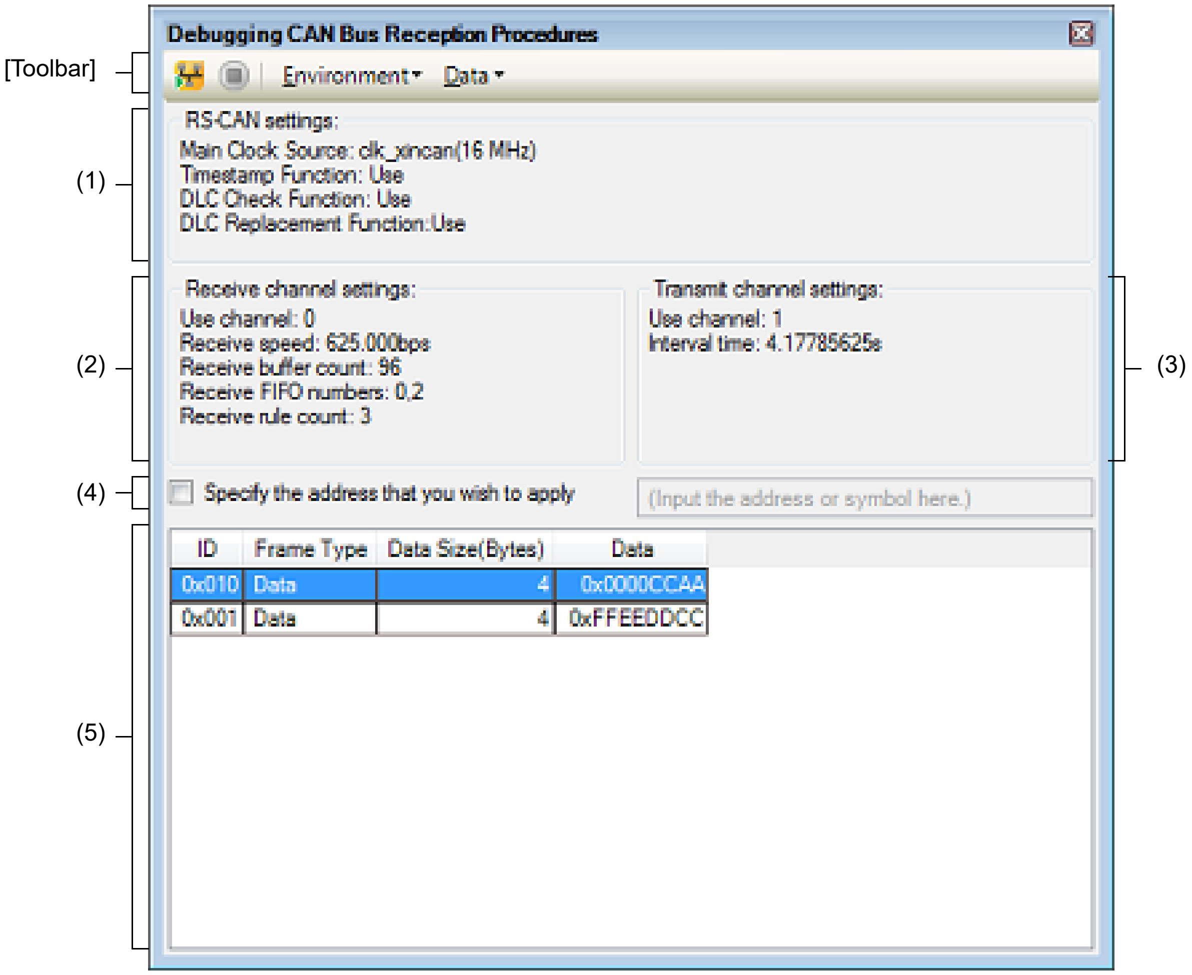
[Function buttons]

Button	Function
OK	Registers the breakpoint set in this dialog box to [Breakpoint List] in the Pseudo-Error Debugging panel [Full-spec emulator][E1][E20].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20]

This panel is central to the functionality of the solution for debugging of CAN bus reception.
See "2.25 Debugging CAN Bus Reception Procedures [Full-spec emulator][E1][E20]" for details on the solution for debugging of CAN bus reception.

Figure A.60 Debugging CAN Bus Reception Procedures Panel



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] >> [Debugging CAN Bus Reception Procedures].
- On the Solution List panel, click the [Debugging CAN Bus Reception Procedures] button.

[Description of each area]

(1) [RS-CAN settings] area
This area displays the following information related to the entire RS-CAN module.

(a) [Main Clock Source]
The source of the clock that is input to the RS-CAN module and its frequency are displayed.

(b) [Timestamp Function]
This displays whether the timestamp function is to be used.

(c) [DLC Check Function]
This displays whether the DLC checking function is to be used.

(d) [DLC Replacement Function]
This displays whether the DLC replacement function is to be used.

Remark Information of the DLC replacement function is displayed only when the DLC checking function is used.

This area only displays the current settings. In order to set new values, open the RS-CAN Module Setting dialog box [Full-spec emulator][E1][E20] from the [RS-CAN Module Setting...] button on the toolbar and edit the values.

(2) [Receive channel settings] area
This area displays the following information related to the receive channel used for debugging the CAN bus reception procedure.

(a) [Use channel]
The number of the channel used as the receive channel is displayed.

(b) [Receive speed]
The receive speed is displayed.

(c) [Receive buffer count]
The number of receive buffers to be used is displayed.

(d) [Receive FIFO numbers]
A list of receive FIFO numbers to be used is displayed.

(e) [Receive rule count]
The number of receive rules that have been set is displayed.

This area only displays the current settings. In order to set new values, open the Receive Channel Setting dialog box [Full-spec emulator][E1][E20] from the [Receive Channel Setting...] button on the toolbar and edit the values.

(3) [Transmit channel settings] area
This area displays the following information related to the transmit channel used for debugging the CAN bus transmission procedure.

(a) [Use channel]
The number of the channel used as the transmit channel is displayed.

(b) [Interval time]
The interval time when continuously transmitting the transmit frames is displayed.

This area only displays the current settings. In order to set new values, open the Transmit Channel Setting dialog box [Full-spec emulator][E1][E20] from the [Transmit Channel Setting...] button on the toolbar and edit the values.

(4) Area for selecting the timing to apply settings
Select the timing to apply the settings made on this panel.

(a) [Specify the address that you wish to apply settings]
Specify whether to apply the settings when the instruction at the specified address has been executed after debugging of CAN bus reception has started.
This text box is valid only when [Specify the address that you wish to apply settings] has been selected.
You can either type an address expression directly into the text box (up to 1024 characters), or select one from the input history via the drop-down list (up to 10 items).

(5) Area for making transmit frame settings
This area lists transmit frames that have already been set.
Each item is explained as follows:

(a) ID
The ID of a transmit frame is displayed as a hexadecimal value.

(b) Frame Type
The type of a transmit frame is displayed. Each frame has either of the following values.

Value	Meaning
Data	Data frame of CAN
Remote	Remote frame of CAN

(c) Data Size(Bytes)
The size of the data in a transmit frame is displayed within the range of 0 to 8 bytes.
"-" is displayed when the frame type is "Remote".

(d) Data
The data of a transmit frame is displayed as a hexadecimal value.
"-" is displayed when the data size is 0 or the frame type is "Remote".

[Toolbar]

	Starts debugging of CAN bus reception based on the settings made in this panel.
	Has the same functionality as the same button on the Debug toolbar in the Main window.
Environment	The following cascade menus are displayed to set up the RS-CAN module.
RS-CAN Module Setting...	Opens the RS-CAN Module Setting dialog box [Full-spec emulator][E1][E20] to make settings for the entire RS-CAN module.
Receive Channel Setting...	Opens the Receive Channel Setting dialog box [Full-spec emulator][E1][E20] to set the receive channel.
Transmit Channel Setting...	Opens the Transmit Channel Setting dialog box [Full-spec emulator][E1][E20] to set the transmit channel.
Data	The following cascade menus are displayed to set transmit frames.
Add Transmit Frame...	Opens the Transmit Frame Setting dialog box [Full-spec emulator][E1][E20] to add a transmit frame.
Edit Transmit Frame...	Opens the Transmit Frame Setting dialog box [Full-spec emulator][E1][E20] to edit the selected transmit frame.
Delete Transmit Frame	Deletes the selected transmit frame.
Export Transmit Frame...	Opens the Save File dialog box to save the transmit frames that have been set in this panel to a CSV file.
Import Transmit Frame...	Opens the Open File dialog box to import the transmit frame settings of this panel from a CSV file. Transmit frames that have already been set will be cleared.

[Context menu]

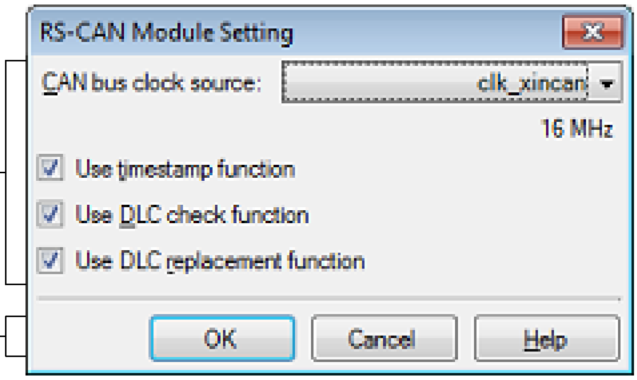
Add Transmit Frame...	Opens the Transmit Frame Setting dialog box [Full-spec emulator][E1][E20] to add a transmit frame.
Edit Transmit Frame...	Opens the Transmit Frame Setting dialog box [Full-spec emulator][E1][E20] to edit the selected transmit frame.
Delete Transmit Frame	Deletes the selected transmit frame.

Export Transmit Frame...	Opens the Save File dialog box to save the transmit frames that have been set in this panel to a CSV file.
Import Transmit Frame...	Opens the Open File dialog box to import the transmit frame settings of this panel from a CSV file. Transmit frames that have already been set will be cleared.

RS-CAN Module Setting dialog box [Full-spec emulator][E1][E20]

This dialog box is used to make settings related to the entire RS-CAN module shown in the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].

Figure A.61 RS-CAN Module Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20], click the [RS-CAN Module Setting...] button on the toolbar.

[Description of each area]

(1) RS-CAN setting area
Set up the entire RS-CAN.

(a) [CAN bus clock source]
Select the clock source for the RS-CAN module from the following drop-down list.

Value	Meaning
clk_xincan	Clock supplied to clk_xincan of RS-CAN
clkc	Clock supplied to clkc of RS-CAN

The actual frequency of the specified clock source appears at the bottom of the drop-down list.

(b) [Use timestamp function]
Specify whether the timestamp function of the RS-CAN is to be used.

(c) [Use DLC check function]
Specify whether the Data Length Code (DLC) checking function of the RS-CAN is to be used.

(d) [Use DLC replacement function]
Specify whether the DLC replacement function of the RS-CAN is to be used.
This item is specifiable only when the DLC checking function is used.

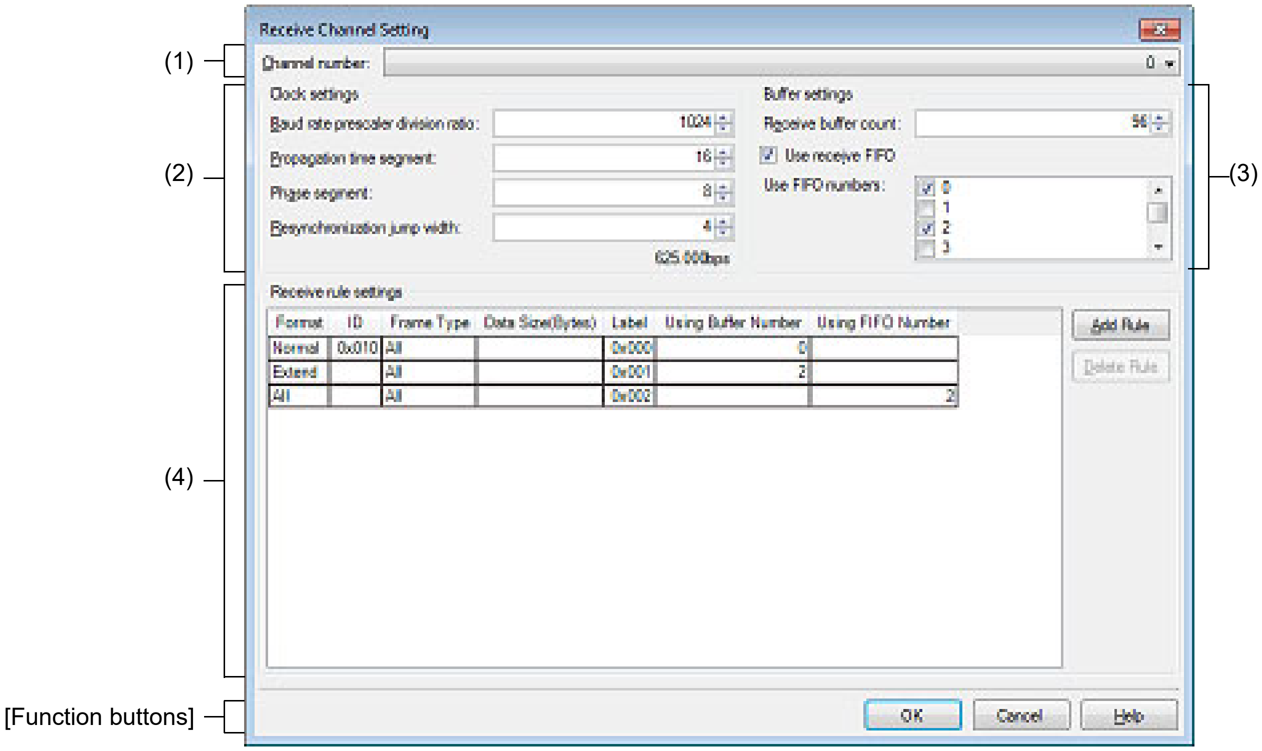
[Function buttons]

Button	Function
OK	Reflects the clock source, timestamp, DLC checking function, and DLC replacement function settings that were made in this dialog box to the [RS-CAN settings] area on the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Receive Channel Setting dialog box [Full-spec emulator][E1][E20]

This dialog box is used to make settings related to the receive channel shown in the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].

Figure A.62 Receive Channel Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20], click the [Receive Channel Setting...] button on the toolbar.

[Description of each area]

(1) [Channel number]
Select the channel for which debugging of CAN bus reception is to be performed.
See the hardware manual for channels that have RS-CAN units.

(2) [Clock settings] area
Set the reception clock to determine the reception speed.
The value of the reception speed is calculated from the following settings and is displayed at the bottom of this area.

(a) [Baud rate prescaler division ratio]
Specify the frequency division ratio of the baud rate prescaler as a decimal value from 1 to 1024.

(b) [Propagation time segment]
Specify the time unit of a propagation time segment as a decimal value from 4 to 16.

(c) [Phase segment]
Specify the time unit of a phase buffer segment as a decimal value from 2 to 8.

(d) [Resynchronization jump width]
Specify the time unit of the resynchronization jump width as a decimal value from 1 to 4.

See "2.25 Debugging CAN Bus Reception Procedures [Full-spec emulator][E1][E20]" for details on calculating the receive speed.

(3) [Buffer settings] area
Make settings related to the receive buffers and receive FIFOs which are used in the receive channel.

(a) [Receive buffer count]
Specify the number of receive buffers as a decimal value from 0 to (Number of channels of units * 16).
If 0 is specified, receive buffers are not used.
Buffer numbers from 0 to (Number of specified buffers - 1) are assigned to receive buffers.

(b) [Use receive FIFO]
Specify whether receive FIFOs are to be used.

(c) [Use FIFO numbers]
This item which displays a list of receive FIFO numbers appears only when the [Use receive FIFO] check box is selected.
Select the receive FIFOs you wish to use from the list.

(4) [Receive rule settings] area
Set the receive rules that are to be applied to the receive channel.

(a) View/edit receive rules
The receive rules that have already been set are listed.
Each item can be selected via the drop-down list or edited directly.
Each item is explained as follows:

<1> Format
Select the format of the frames to be received from the following drop-down list.

Value	Meaning
Normal	Only frames of the normal format are received.
Extend	Only frames of the extended format are received.
All	The frames are not sorted by the format, and frames of all formats are received.

<2> ID
Specify the ID values of the frames to be received as hexadecimal values.
Only the frames that match the specified ID values are received. If this item is blank, frames of all ID values are received.

Remark 1. The specifiable range of the hexadecimal value for the ID value depends on the format as shown below.

Format	Settable Range
Normal	0x0 to 0x7FF
Extend	0x0 to 0x1FFFFFFF
All	

Remark 2. When the value of the format was changed to "Extend" or changed from "All" to "Normal", a value filtered by 0x7FF is automatically set.

<3> Frame Type
Select the type of the frames to be received from the following drop-down list.

Value	Meaning
Data	Only data frames are received.

Value	Meaning
Remote	Only remote frames are received.
All	The frames are not sorted by the frame, and all frames are received.

<4> Data Size(Bytes)
Specify the data size (bytes) of each frame to be received within the range of 1 to 8 bytes.
If this is blank, frames of all data sizes are received.

Remark When the value of the frame type was changed to "Data" or changed from "All" to "Remote", this item is left blank automatically.

<5> Label
Specify the label value to be added to the frames that have passed the conditions of <1> to <4>.
The label value should be a hexadecimal value from 0 to 0xFFF.

<6> Using Buffer Number
Specify the receive buffer number for storing the frames that have passed the conditions of <1> to <4>.
The following values can be specified.

Value	Meaning
Decimal value from 0 to ([Receive buffer count] setting - 1)	Frames are stored in the buffer with the specified receive buffer number.
(Blank)	No receive buffer is used.

Remark When the value of the storage FIFO number is changed from blank to a number, this item is made blank automatically.

<7> Using FIFO Number
Specify the receive FIFO number for storing the frames that have passed the conditions of <1> to <4>.
The following values can be specified.

Value	Meaning
Decimal value of [Use FIFO numbers] that can be used	Frames are stored in the FIFO with the specified receive FIFO number.
(Blank)	No receive FIFO is used.

Remark When the value of the storage buffer number is changed from blank to a number, this item is made blank automatically.

(b) Default values of receive rules
The default values of receive rules are as follows:

Format	ID	Frame Type	Data Size (Bytes)	Label	Using Buffer Number	Using FIFO Number
All	(Blank)	All	(Blank)	0x0x	(Blank)	(Blank)

(c) Button

Add Rule	Adds a new receive rule to the receive rule list. A newly added rule has the default values (see "(b) Default values of receive rules").
Delete Rule	Deletes the receive rule selected in the receive rule list.

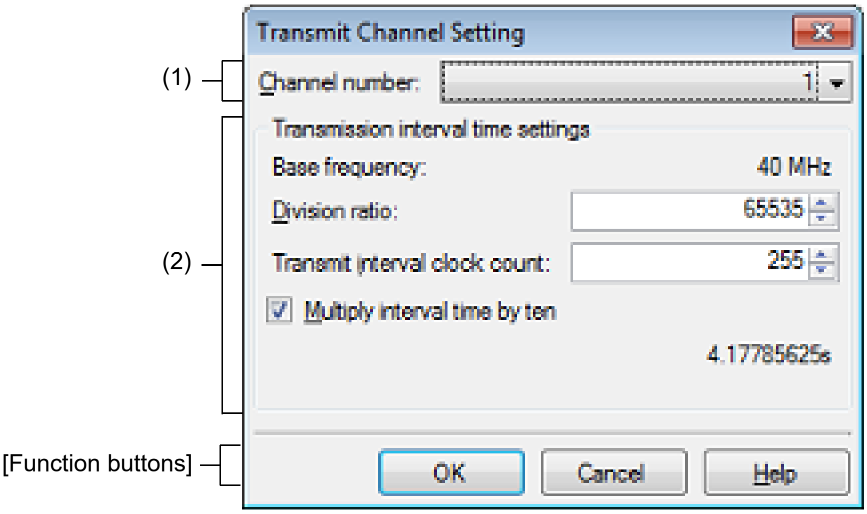
[Function buttons]

Button	Function
OK	Reflects the settings of the receive channel that were set in this dialog box to the [Receive channel settings] area on the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Transmit Channel Setting dialog box [Full-spec emulator][E1][E20]

This dialog box is used to make settings related to the transmit channel shown in the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].

Figure A.63 Transmit Channel Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20], click the [Transmit Channel Setting...] button on the toolbar.

[Description of each area]

(1) [Channel number]
Select the channel used to transmit frames for debugging of CAN bus reception.
See the hardware manual for channels that have RS-CAN units.

(2) [Transmission interval time settings] area
Make settings related to the interval time for transmitting frames in debugging of CAN bus reception.
The value of the interval time is calculated from the following settings and is displayed at the bottom of this area.

(a) [Base frequency]
The base frequency of the timer for measuring the continuous transmission interval of the RS-CAN is displayed.

Remark Half of the frequency of pclk which is input to the RS-CAN is used as the base frequency for debugging the CAN bus reception procedure.

(b) [Division ratio]
Set a decimal value from 1 to 65535 as the division ratio of the base frequency.

(c) [Transmit interval clock count]
Set a decimal value from 1 to 255 as the value to divide the clock specified in (b) and the result becomes the interval for the clock used in transmission.

(d) [Multiply interval time by ten]
Select whether to multiply the interval time set in (b) and (c) by 10.

See "2.25 Debugging CAN Bus Reception Procedures [Full-spec emulator][E1][E20]" for details on calculating the interval time of continuous transmission.

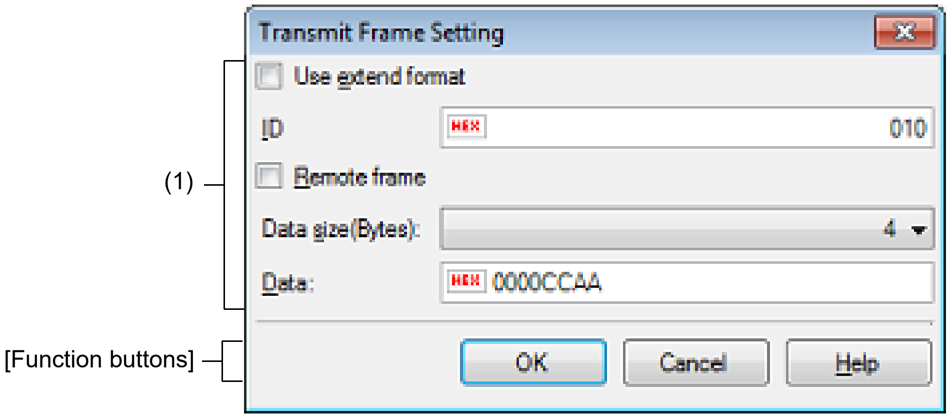
[Function buttons]

Button	Function
OK	This dialog box is used to make settings related to the transmit frame shown in the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Transmit Frame Setting dialog box [Full-spec emulator][E1][E20]

This dialog box is used to make settings related to the transmit frame shown in the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].

Figure A.64 Transmit Frame Setting Dialog Box



This section describes the following.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20], select [Add Transmit Frame...] from the context menu.
- On the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20], select [Edit Transmit Frame...] from the context menu.

[Description of each area]

(1) Transmit frame setting area
Make settings related to the transmit frame.

(a) [Use extend format]
The RS-CAN supports both the normal format and extended format for a CAN bus frame.
Select whether to use the extended format for the frame to be transmitted.

(b) [ID]
Directly enter the ID of a CAN bus frame into the text box as a hexadecimal value.
The range of values that can be input varies depending on the format as shown below.

Format	Settable Range
Normal	0x0 to 0x7FF
Extend	0x0 to 0x1FFFFFFF

(c) [Remote frame]
The feature of debugging the CAN bus reception procedure supports only debugging of data frames and remote frames of the CAN bus.
Select whether to transmit a remote frame.
If this check box is selected, the frame is assumed to be a remote frame and data cannot be set in the frame.

(d) [Data size(Bytes)]
Select the size of a data frame of the CAN bus within the range of 0 to 8 bytes.

(e) [Data]
Directly enter the data to be transmitted in a data frame of the CAN bus into the text box as a hexadecimal value.
The range of the data that can be input is the range up to the size specified in [Data size(Bytes)].

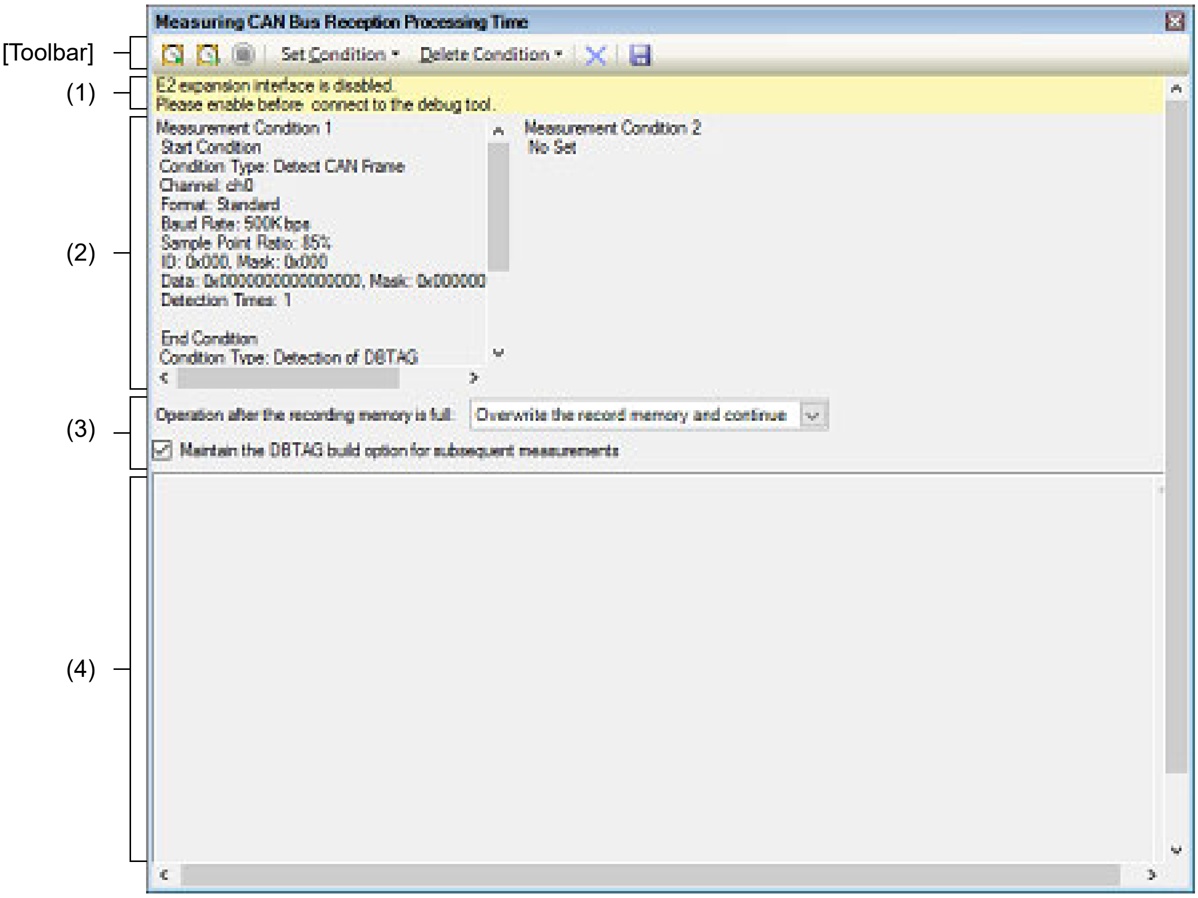
[Function buttons]

Button	Function
OK	Reflects the settings of a transmit frame that were set in this dialog box to the area for making transmit frame settings on the Debugging CAN Bus Reception Procedures panel [Full-spec emulator][E1][E20].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Measuring CAN Bus Reception Processing Times panel [E2]

This panel is central to the functionality of the solution for measurement of CAN bus reception processing times.
See "2.26 Measuring CAN Bus Reception Processing Times [E2]" for details on the solution for measurement of CAN bus reception processing times.

Figure A.65 Measuring CAN Bus Reception Processing Times Panel



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] >> [Measure Reception Processing Time].
- On the Solution List panel, click the [Measuring CAN Bus Reception Processing Times] button.

[Description of each area]

(1) Warning display area
A warning will appear in this area when the current settings for measuring CAN bus reception processing times do not satisfy the conditions required prior to connection of the debug tool. This area is hidden when there is no warning to be reported.

(2) Measurement conditions display area
This area displays the conditions for measuring CAN bus reception processing times.
Note that you can only view the current settings in this area. To make settings, use the Measurement Condition Setting dialog box [E2] opened by selecting [Set Condition] -> [Set Condition *n*...].

Remark You can select up to two conditions.
Conditions 1 and 2 that have been set are displayed in order (from the left) in this area.

(3) General settings area
This area is used to make general settings that will apply to all measurements, regardless of conditions 1 and 2.

(a) [Operation after the recording memory is full]
Select the operation of the debugger after the recording memory of the emulator becomes full during measurement from the following drop-down list.

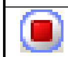
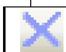
Overwrite the record memory and continue	Recording of trace data and sampled CAN data are continued by overwriting the oldest data.
Stop recording	The output of software trace data and sampled CAN data stop. Note that execution of the program will not stop.
Stop program	Execution of the program and the output of software trace data and sampled CAN data stop.

(b) [Maintain the DBTAG build option for subsequent measurements]
Select this check box if you have built a program with the inclusion of automatically inserted dbtag instructions and wish to apply this build option to the given property of the build tool.

Caution This check box is selectable when V1.06.00 or later of the CC-RH compiler is selected for the active project.

(4) Measurement results display area
This area displays the results of measuring CAN bus reception processing times (minimum time, maximum time, average time, and number of measurement).

[Toolbar]

	Starts measuring the CAN bus reception processing time. When V1.06.00 or later of the CC-RH compiler is selected for the active project, prior to measurement, rebuilding to insert dbtag instructions and downloading of the result proceed.
	Starts measuring the CAN bus reception processing time. Rebuilding to insert dbtag instructions and downloading of the result do not proceed. Note that this button appears when V1.06.00 or later of the CC-RH compiler is selected for the active project.
	Has the same functionality as the same button on the Debug toolbar in the Main window.
Set Condition	The following cascade menus are displayed to set conditions for measurement of the CAN bus reception processing time.
Set Condition 1...	Opens the Measurement Condition Setting dialog box [E2] to make settings for measurement condition 1.
Set Condition 2...	Opens the Measurement Condition Setting dialog box [E2] to make settings for measurement condition 2.
Delete Condition	The following cascade menus are displayed to delete the delete the condition for measurement of the CAN bus reception processing time.
Delete Condition 1	Deletes measurement condition 1.
Delete Condition 2	Deletes measurement condition 2.
	Clears the log of measurement results of the CAN bus reception processing time.

	Opens the Save File dialog box to save measurement results of the CAN bus reception processing time to a CSV file or Microsoft Office Excel book.

[Context menu]

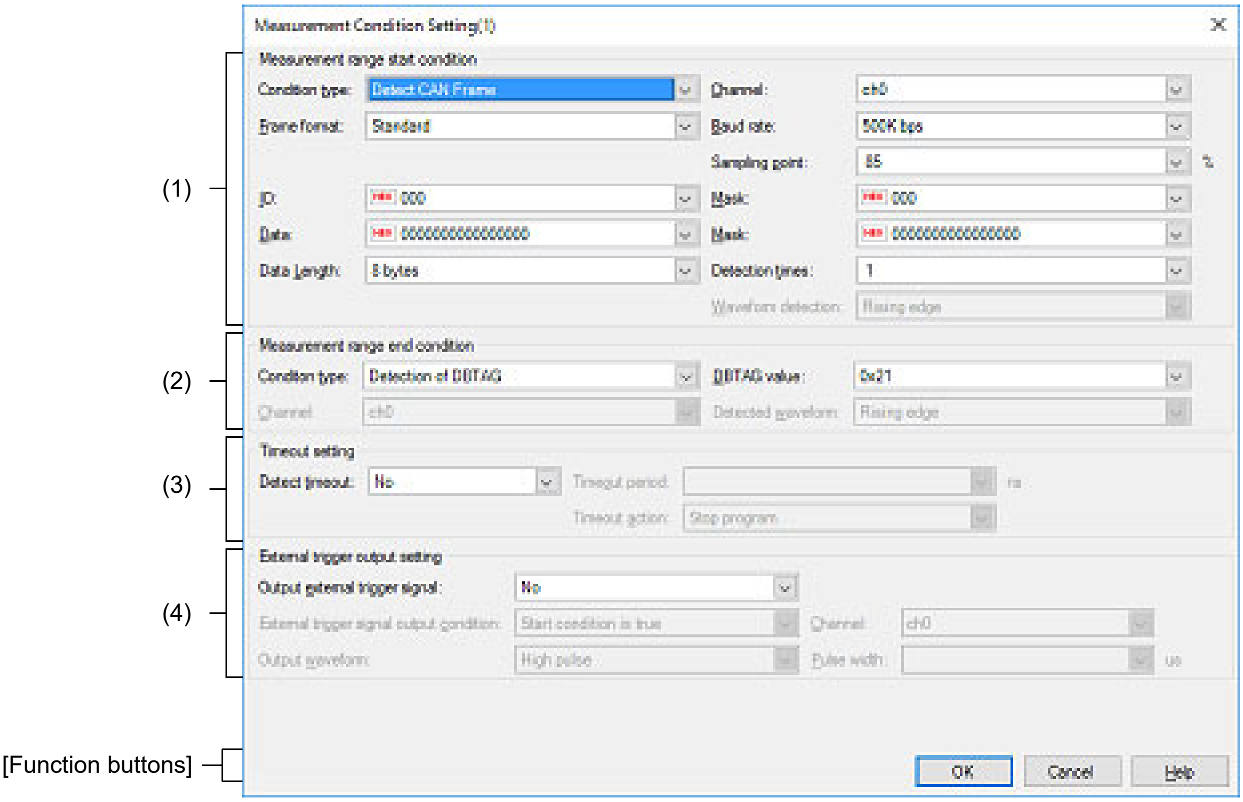
Set Condition	The following cascade menus are displayed to set conditions for measurement of the CAN bus reception processing time.
Set Condition 1...	Opens the Measurement Condition Setting dialog box [E2] to make settings for measurement condition 1.
Set Condition 2...	Opens the Measurement Condition Setting dialog box [E2] to make settings for measurement condition 2.
Delete Condition	The following cascade menus are displayed to delete the delete the condition for measurement of the CAN bus reception processing time.
Delete Condition 1	Deletes measurement condition 1.
Delete Condition 2	Deletes measurement condition 2.
Clear	Clears the log of measurement results of the CAN bus reception processing time.

Measurement Condition Setting dialog box [E2]

This dialog box is used to set conditions for measurement in the Measuring CAN Bus Reception Processing Times panel [E2].

You can make settings for the measurement condition with the number you have selected before opening this dialog box.

Figure A.66 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 CAN Bus Reception Processing Times panel [E2], click [Set Condition 1...] or [Set Condition 2...] from the [Set Condition] button on the toolbar.

[Description of each area]

(1) [Measurement range start condition] area

This area allows you to select conditions that define the beginning of the range over which measurement of CAN bus reception processing times will proceed.

(a) [Condition type]

Select "Detect CAN Frame" or "Detect External Trigger Input Signal" as the type of the condition.

(b) [Channel]

Select ch0 or ch1 as the channel number to be detected, that is, the number of the CAN channel or external trigger input channel depending on whether "Detect CAN Frame" or "Detect External Trigger Input Signal" has been selected for [Condition type].

(c) [Frame format]
If you have selected "Detect CAN Frame" for [Condition type], select "Standard" or "Extended" as the format of CAN frames to be detected.

(d) [Baud rate]
If you have selected "Detect CAN Frame" for [Condition type], select one of the following values as the bit rate for use in the detection of CAN communications.
1M bps (default), 500K bps, 250K bps, 125K bps

(e) [Sampling point]
If you have selected "Detect CAN Frame" for [Condition type], specify the relative position as a percentage within one bit period for the sampling of each bit of data in CAN frames to be detected. Decimal values from 1 to 100 can be specified.

(f) [ID], [Mask]
If you have selected "Detect CAN Frame" for [Condition type], specify ID in CAN frames to be detected and the mask value to be used as hexadecimal values.
The range of values that can be specified varies depending on selection of [Frame format] as shown below.

Selection of [Frame format]	Settable Range
Standard	0 to 7FF
Extended	0 to 1FFFFFFF

When 0 is selected for a mask bit, the bit is treated as being masked.
If [Mask] is blank, all current ID bits are treated as not being masked.

(g) [Data], [Mask]
If you have selected "Detect CAN Frame" for [Condition type], specify data in CAN frames to be detected and the mask value to be used as hexadecimal values in the range from 0 to FFFFFFFFFFFFFFFF.
The values specified in the sequence are assumed to be in data fields 0, 1, 2, ..., in that order, within the CAN frames. When the value of the last digit takes up less than one byte, the lower-order bits are padded with 0.

Example When the input value is 0011223, it is assumed that 0x00 is to be found in data field 0, 0x11 in data field 1, 0x22 in data field 2, and 0x30 in data field 3.

When 0 is selected for a mask bit, the bit is treated as being masked.
If [Mask] is blank, all current ID bits are treated as not being masked.

(h) [Data Length]
If you have selected "Detect CAN Frame" for [Condition type], select the data size in bytes for CAN frames to be detected as a value from 0 to 8.

(i) [Detection times]
If you have selected "Detect CAN Frame" for [Condition type], time measurement starts when CAN frames have been detected the number of times specified in this field.

(j) [Waveform detection]
If you have selected "Detect External Trigger Input Signal" for [Condition type], select the type of the external trigger input waveform to be detected from the following drop-down list.

Rising edge	Rising edges are detected.
Falling edge	Falling edges are detected.
Both edges	Both rising and falling edges are detected.

(2) [Measurement range end condition] area
This area allows you to select conditions that define the end of the range over which measurement of CAN bus reception processing times will proceed.

(a) [Condition type]
Select "Detection of DBTAG" or "Detection of external input trigger signal" as the type of the condition.

(b) [DBTAG value]
If you have selected "Detection of DBTAG" for [Condition type], select the value of DBTAG to be detected.
The facility for measuring CAN bus reception processing times supports the following ten values.

0x21, 0x29, 0x31, 0x39, 0x41, 0x49, 0x51, 0x59, 0x61, 0x69

(c) [Channel]
If you have selected "Detection of external input trigger signal" for [Condition type], select ch0 or ch1 as the channel number to be detected.

(d) [Detected waveform]
If you have selected "Detection of external input trigger signal" for [Condition type], select the type of the external trigger input waveform to be detected from the following drop-down list.

Rising edge	Rising edges are detected.
Falling edge	Falling edges are detected.
Both edges	Both rising and falling edges are detected.

(3) [Timeout setting] area
This area allows you to make timeout settings for the measurement of CAN bus reception processing times.

(a) [Detect timeout]
Select whether to detect timeout.

(b) [Timeout period]
Enter a decimal value from 0 to 2,345,624,805,922,133 (in nanoseconds) as the timeout value.

(c) [Timeout action]
Select the action on timeout detection from the following drop-down list.

Detection only	Detection of timeout is only used as a condition for external trigger output.
Stop internal tracing	Tracing within the MCU stops but execution of the program continues.
Stop program	Execution of the program stops.

Caution "Stop internal tracing" is not selectable as the action on timeout detection when "Detection of DBTAG" is selected as the type of measurement-end condition.

(4) [External trigger output setting] area
This area allows you to make settings for the output of external trigger signals in cases where measuring CAN bus reception processing times needs to work with an external device.

(a) [Output external trigger signal]
Select whether to output an external trigger signal.

(b) [External trigger signal output condition]
Select the timing for the output of an external trigger signal from the following drop-down list.

Start condition is true	An external trigger signal will be output when the condition set in the [Measurement range start condition] area is satisfied.
End condition is true	An external trigger signal will be output when the condition set in the [Measurement range end condition] area is satisfied.
Timeout condition is true	An external trigger signal will be output when the condition set in the [Timeout setting] area is satisfied.

Caution "Start condition is true" is not selectable when "Detect External Trigger Input Signal" is selected as the type of measurement-start condition.
"End condition is true" is also not selectable when "Detect External Trigger Input Signal" is selected as the type of measurement-end condition.
"Timeout condition is true" is not selectable when timeout detection is disabled.

(c) [Channel]
Select ch0 or ch1 as the channel from which an external trigger signal will be output.

(d) [Output waveform]
The waveform of the pulse signal to be output as the external trigger is displayed.

(e) [Pulse width]
Specify the width of the pulse signal to be output as the external trigger as a decimal value from 1 to 65535 (in microseconds).

[Function buttons]

Button	Function
OK	Clicking on this button applies the conditions set in this dialog box as measurement conditions in the Measuring CAN Bus Reception Processing Times panel [E2].
Cancel	Nullifies settings and closes this dialog box.
Help	Displays the help for this dialog box.

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Nov 01, 2024	-	First Edition issued

CS+ V8.13.00 User's Manual:
RH850 Debug Tool

Publication Date: Rev.1.00 Nov 01, 2024
Published by: Renesas Electronics Corporation

CS+ V8.13.00

Renesas Electronics Corporation