Integrated Development Environment for RX Family

Migration to New Integrated Development Environment “CubeSuite+”: Emulator

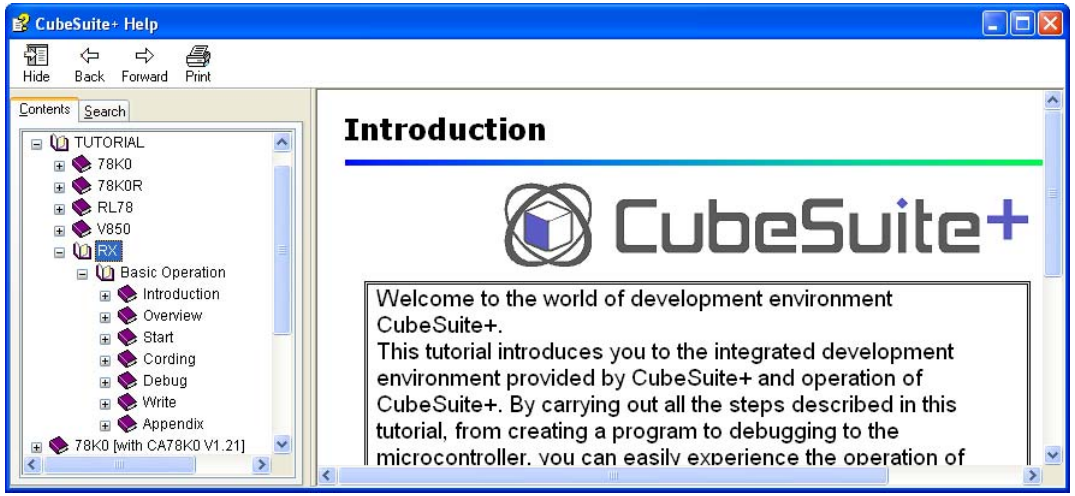
Renesas Electronics Corporation
MCU Business Unit
MCU Software Division
MCU Tool Product Marketing Department

February 13, 2012 Revision 1.01

R20UT2056EJ0101

Introduction

This document describes how to migrate the High-performance Embedded Workshop for RX Family to CubeSuite+ and how to operate E1 and E20 emulators in the CubeSuite+ environment, on the basis of CubeSuite+ V1.01.00.
For toolchains, refer to *Integrated Development Environment for RX Family Migration to Integrated Development Environment “CubeSuite+”: Build*.
Also refer to the tutorial guide provided by CubeSuite+ for how to use tools.
The tutorial guide is available by selecting [Help] -> [Tutorial] from the CubeSuite+ menu.



Welcome to the world of development environment CubeSuite+.
This tutorial introduces you to the integrated development environment provided by CubeSuite+ and operation of CubeSuite+. By carrying out all the steps described in this tutorial, from creating a program to debugging to the microcontroller, you can easily experience the operation of

Tutorial Guide

Contents

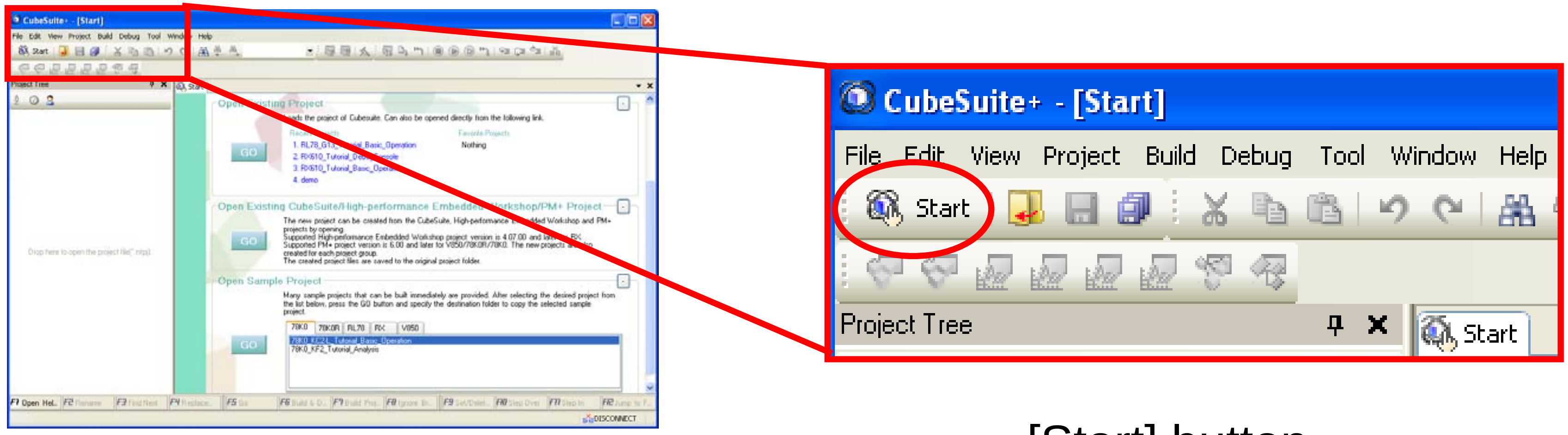
1. Opening a Workspace of High-performance Embedded Workshop by using CubeSuite+
2. Changing the Debugger (E1/E20 Emulator or Simulator)
3. Changing the MCU
4. Where Do We Make Settings when Connecting an Emulator?
5. Connecting an Emulator
6. Disconnecting the Emulator
7. Starting an Emulator with Hot Plug-in
8. Entering an ID Code
9. Downloading a Program
10. Registering Additional Download Files
11. Starting/Stopping a Program
12. Setting the Start/Stop Function
13. Setting Trace Acquisition Conditions
14. Setting Trace Start/Stop Conditions
15. Setting Trace Extraction Conditions
16. Viewing/Changing Memory Data and Variables While the Program is Running

Contents

1. Opening a Workspace of High-performance Embedded Workshop by CubeSuite+

The CubeSuite+ can open a workspace created by the High-performance Embedded Workshop. Perform the following procedure to open a workspace.

(1) Click the [Start] button at the upper left of the toolbar to open the start panel.


CubeSuite+ - [Start]
File Edit View Project Build Debug Tool Window Help
Start
Project Tree
Open Existing Project
GO
Recent Projects
Favorite Projects
Open Existing CubeSuite/High-performance Embedded Workshop/PM+ Project
Open Sample Project
78K0 78K0R RL78 RX V850
Drop here to open the project file(*.mtpj)
F1 Open Hel.. F2 Rename F3 Find Next F4 Replace... F5 Go F6 Build & D... F7 Build Pro... F8 Ignore Br... F9 Set/Delet... F10 Step Over F11 Step In F12 Jump to F...
DISCONNECT

Loads the project of Cubesuite. Can also be opened directly from the following link.

1. RL78_G13_Tutorial_Basic_Operation
2. RX610_Tutorial_Debug_Console
3. RX610_Tutorial_Basic_Operation
4. demo

Nothing

The new project can be created from the CubeSuite, High-performance Embedded Workshop and PM+ projects by opening.
Supported High-performance Embedded Workshop project version is 4.07.00 and later for RX.
Supported PM+ project version is 6.00 and later for V850/78K0R/78K0. The new projects are also created for each project group.
The created project files are saved to the original project folder.

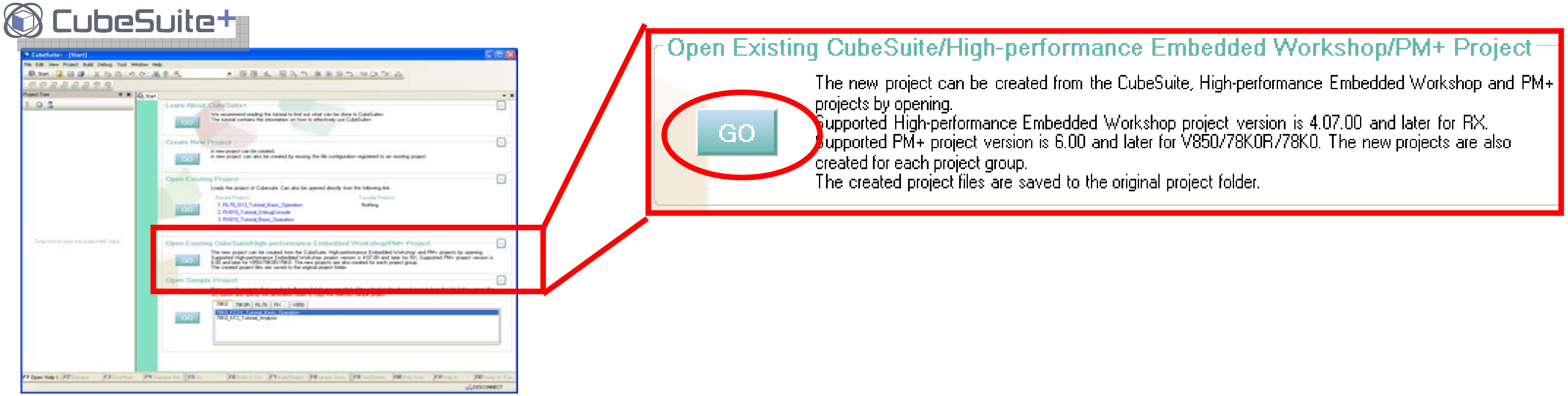
Many sample projects that can be built immediately are provided. After selecting the desired project from the list below, press the GO button and specify the destination folder to copy the selected sample project.

78K0_KC2-L_Tutorial_Basic_Operation
78K0_KF2_Tutorial_Analysis

[Start] button

1. Opening Workspace of High-performance Embedded Workshop by Using CubeSuite+

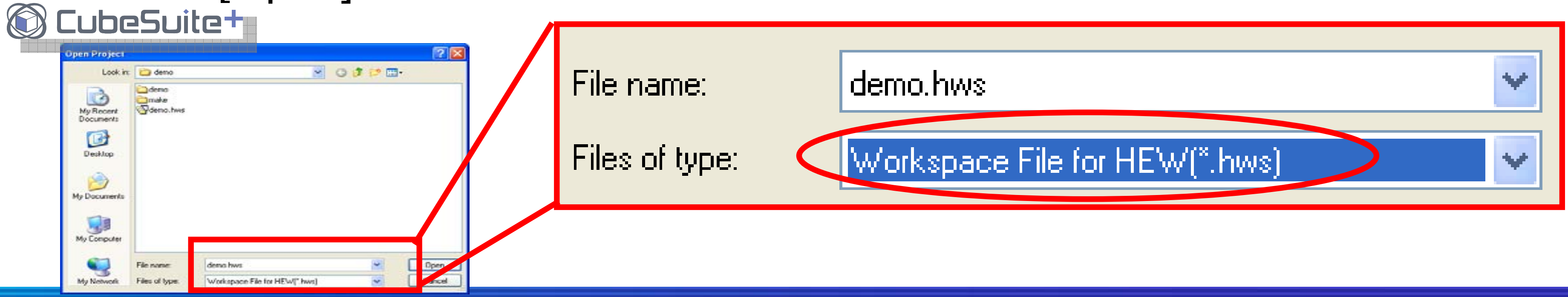
(2) Click the [GO] button to the left of [Open Existing CubeSuite/High-performance Embedded Workshop/PM+ Project].



Open Existing CubeSuite/High-performance Embedded Workshop/PM+ Project

The new project can be created from the CubeSuite, High-performance Embedded Workshop and PM+ projects by opening.
Supported High-performance Embedded Workshop project version is 4.07.00 and later for RX.
Supported PM+ project version is 6.00 and later for V850/78K0R/78K0. The new projects are also created for each project group.
The created project files are saved to the original project folder.

(3) The [Open Project] dialog box opens. Select [Workspace File for HEW(*.hws)] for file type, specify a workspace created by the HEW, and click the [Open] button.

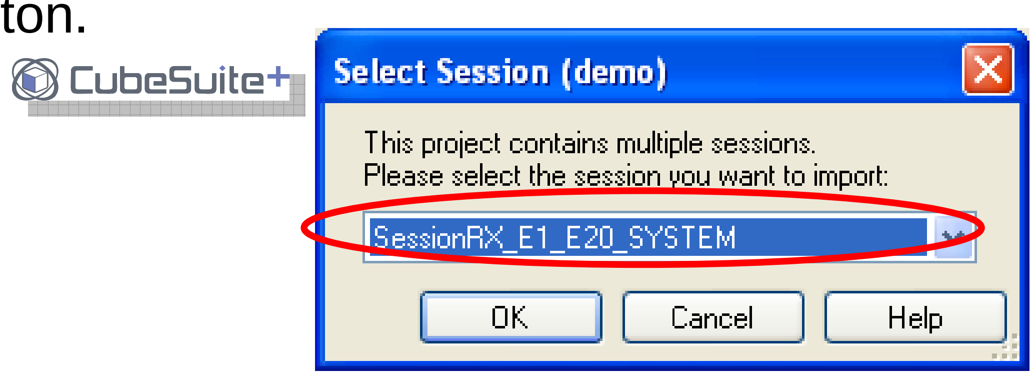


File name: demo.hws

Files of type: Workspace File for HEW(*.hws)

1. Opening Workspace of High-performance Embedded Workshop by Using CubeSuite+

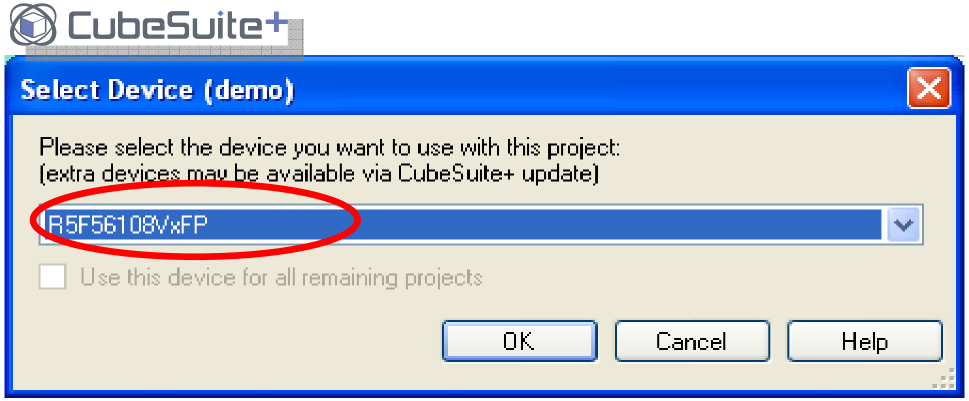
(4) If the HEW workspace contains multiple sessions, the [Select Session] dialog box opens. Select the session you want to use and click the [OK] button.



This project contains multiple sessions.
Please select the session you want to import:

SessionRX_E1_E20_SYSTEM

(5) Then the [Select Device] dialog box opens. Select the name of the device you want to use and click the [OK] button.
The HEW workspace is converted to a CubeSuite+ project.



Please select the device you want to use with this project:
(extra devices may be available via CubeSuite+ update)

R5F56108VxFP

□ Use this device for all remaining projects

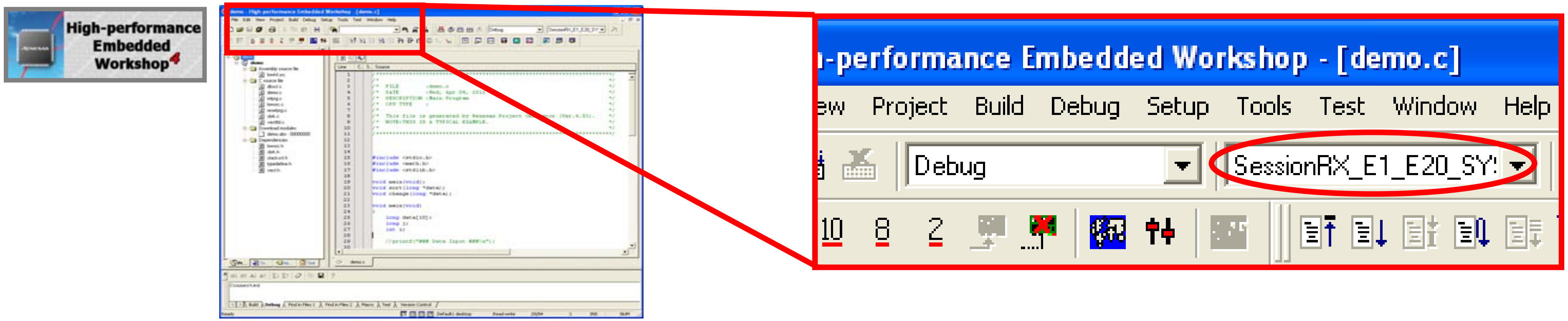
Caution

Once a device has been set, it can no longer be changed.
For how to change the device, see "3. Changing the MCU."

2. Changing the Debugger (E1/E20 Emulator or Simulator)

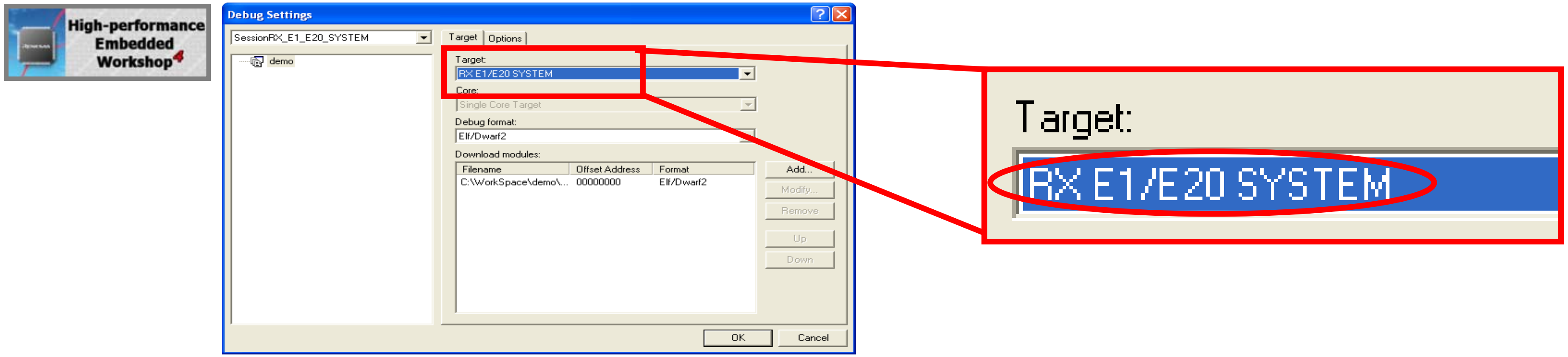
The HEW allowed users to select a debugger (E1/E20 emulator or simulator) in the process of changing the debug session or the target shown in the [Debug Settings] dialog box.

The CubeSuite+, on the other hand, allows users to select a debugger on the project tree. The procedure to change the debugger is described on the following pages.


High-performance Embedded Workshop4
-performance Embedded Workshop - [demo.c]
ew
Project
Build
Debug
Setup
Tools
Test
Window
Help
10
8
2

Debug

SessionRX_E1_E20_SY!



SessionRX_E1_E20_SYSTEM

Target: RX E1/E20 SYSTEM

Core: Single Core Target

Debug format: Elf/Dwarf2

Download modules:

Filename	Offset Address	Format
C:\WorkSpace\demo\...	00000000	Elf/Dwarf2

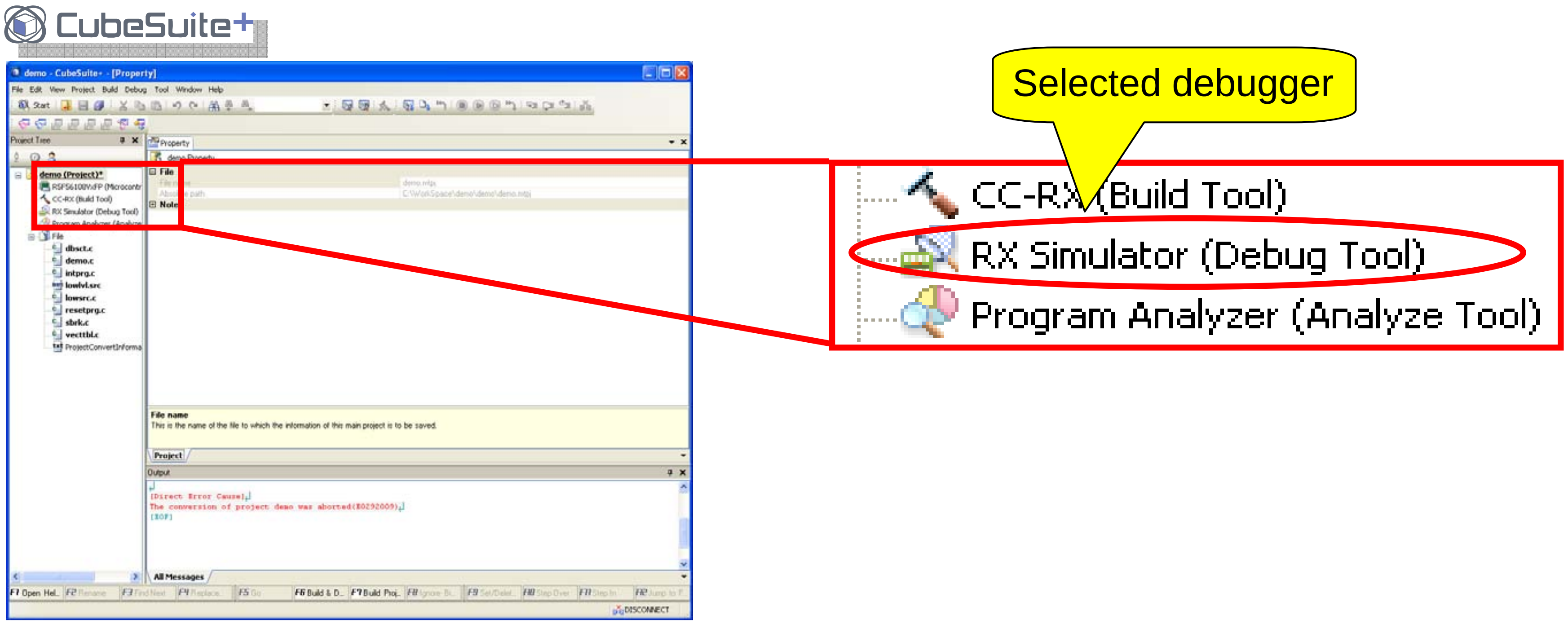
Target:

RX E1/E20 SYSTEM

Selecting a Debugger (E1/E20 Emulator or Simulator) in the HEW

2. Changing the Debugger (E1/E20 Emulator or Simulator)

(1) The debug tool name (debug tool) on the project tree panel indicates the currently selected debugger.
The following example shows that the RX simulator is selected:


CubeSuite+
demo - CubeSuite+ - [Property]
File Edit View Project Build Debug Tool Window Help
Start
Project Tree
demo (Project)*
R5F56108VxFP (Microcontr
CC-RX (Build Tool)
RX Simulator (Debug Tool)
Program Analyzer (Analyze
File
dbsct.c
demo.c
intprg.c
lowlvl.src
lowsrc.c
resetprg.c
sbrk.c
vecttbl.c
ProjectConvertInforma
Property
demo Property
File
File name
Absolute path
Note
Project
Output
All Messages
F1 Open Hel.. F2 Rename F3 Find Next F4 Replace.. F5 Go F6 Build & D.. F7 Build Proj.. F8 Ignore Br.. F9 Set/Delet.. F10 Step Over F11 Step In F12 Jump to F..
DISCONNECT
Selected debugger
CC-RX (Build Tool)
RX Simulator (Debug Tool)
Program Analyzer (Analyze Tool)

File name: demo.mtpj

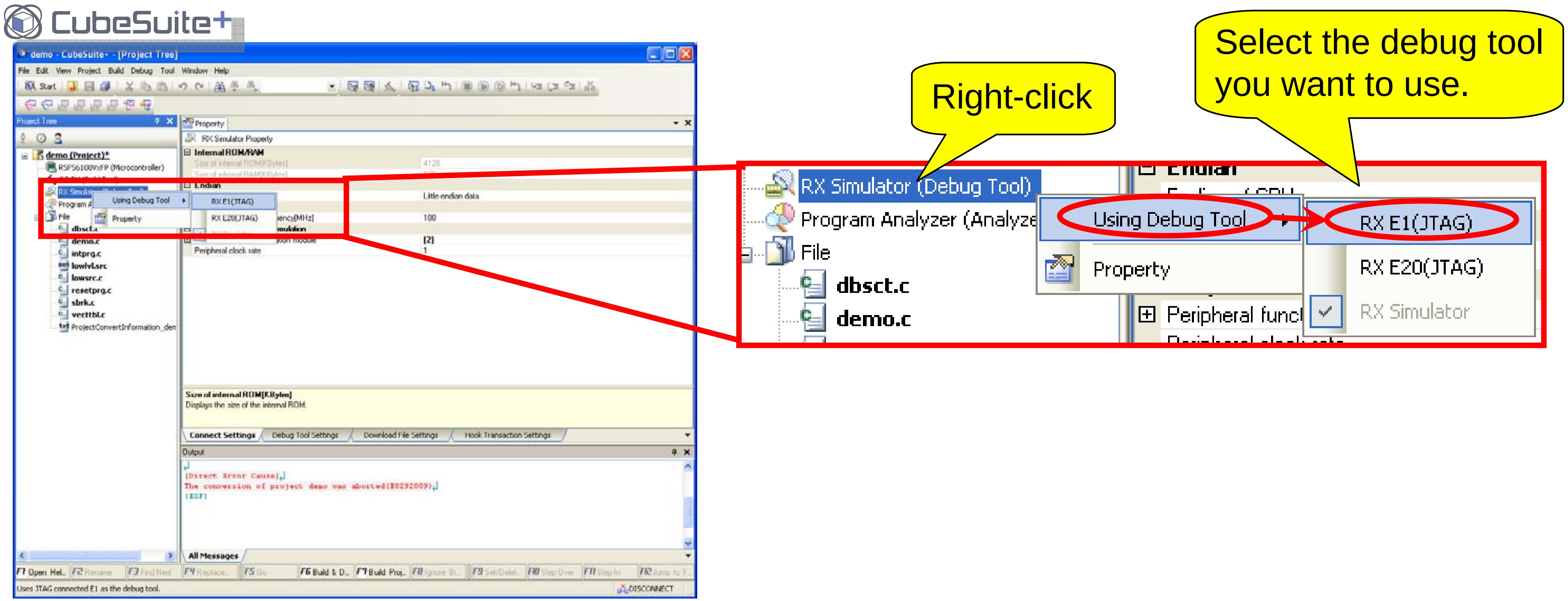
Absolute path: C:\WorkSpace\demo\demo\demo.mtpj

File name
This is the name of the file to which the information of this main project is to be saved.

```
[Direct Error Cause]
The conversion of project demo was aborted(E0292009)
[EOF]
```

2. Changing the Debugger (E1/E20 Emulator or Simulator)

(2) To change the debugger, right-click the debug tool name (debug tool) to open a pop-up menu. Select [Using Debug Tool] from the pop-up menu to select the debug tool you want to use.



CubeSuite+
demo - CubeSuite+ - [Project Tree]
File Edit View Project Build Debug Tool Window Help
Project Tree
demo (Project)*
R5F56108VxFP (Microcontroller)
RX Simulator (Debug Tool)
Program Analyzer (Analyze
File
dbsct.c
demo.c
intprg.c
lowlvl.src
lowsrc.c
resetprg.c
sbrk.c
vecttbl.c
ProjectConvertInformation_dem
Using Debug Tool
Property
RX E1(JTAG)
RX E20(JTAG)
RX Simulator
Property
RX Simulator Property
Internal ROM/RAM
Endian
Peripheral function
Peripheral clock rate
Connect Settings
Debug Tool Settings
Download File Settings
Hook Transaction Settings
Output
All Messages
F1 Open Hel.. F2 Rename F3 Find Next F4 Replace.. F5 Go F6 Build & D.. F7 Build Proj.. F8 Ignore Br.. F9 Set/Delet.. F10 Step Over F11 Step In F12 Jump to F..
DISCONNECT

Size of internal ROM[KBytes] 4120

Endian: Little endian data

Peripheral clock rate 1

Size of internal ROM[KBytes]
Displays the size of the internal ROM.

```
[Direct Error Cause]
The conversion of project demo was aborted(E0292009).
[EOF]
```

Uses JTAG connected E1 as the debug tool.

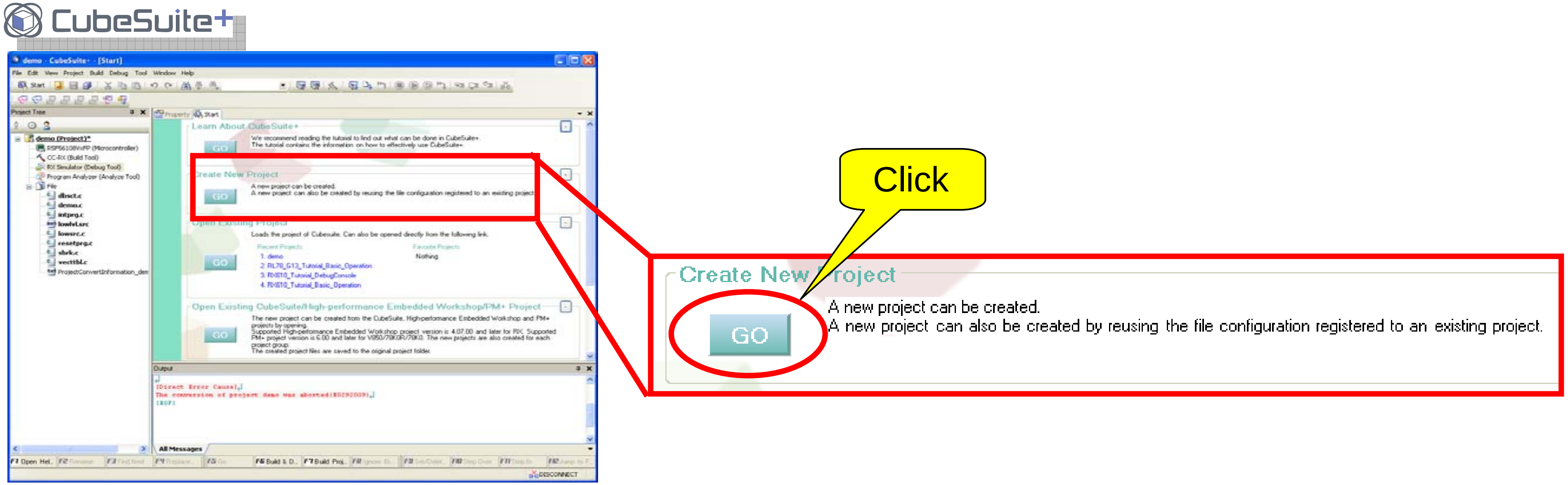
Right-click

Select the debug tool you want to use.

3. Changing the MCU

The CubeSuite+ does not allow changing of the MCU for a project that has already been created.
To change the MCU, a new project must be created based on an existing project.
Perform the following procedure to create a project:

(1) Click the [Start] button at the upper left of the toolbar to open the start panel.
(2) Click the [GO] button on the left of the [Create New Project] field.



Learn About CubeSuite+

We recommend reading the tutorial to find out what can be done in CubeSuite+.
The tutorial contains the information on how to effectively use CubeSuite+.

Create New Project

A new project can be created.
A new project can also be created by reusing the file configuration registered to an existing project.

Open Existing Project

Loads the project of Cubesuite. Can also be opened directly from the following link.

1. demo
2. RL78_G13_Tutorial_Basic_Operation
3. RX610_Tutorial_DebugConsole
4. RX610_Tutorial_Basic_Operation

Nothing

Open Existing CubeSuite/High-performance Embedded Workshop/PM+ Project

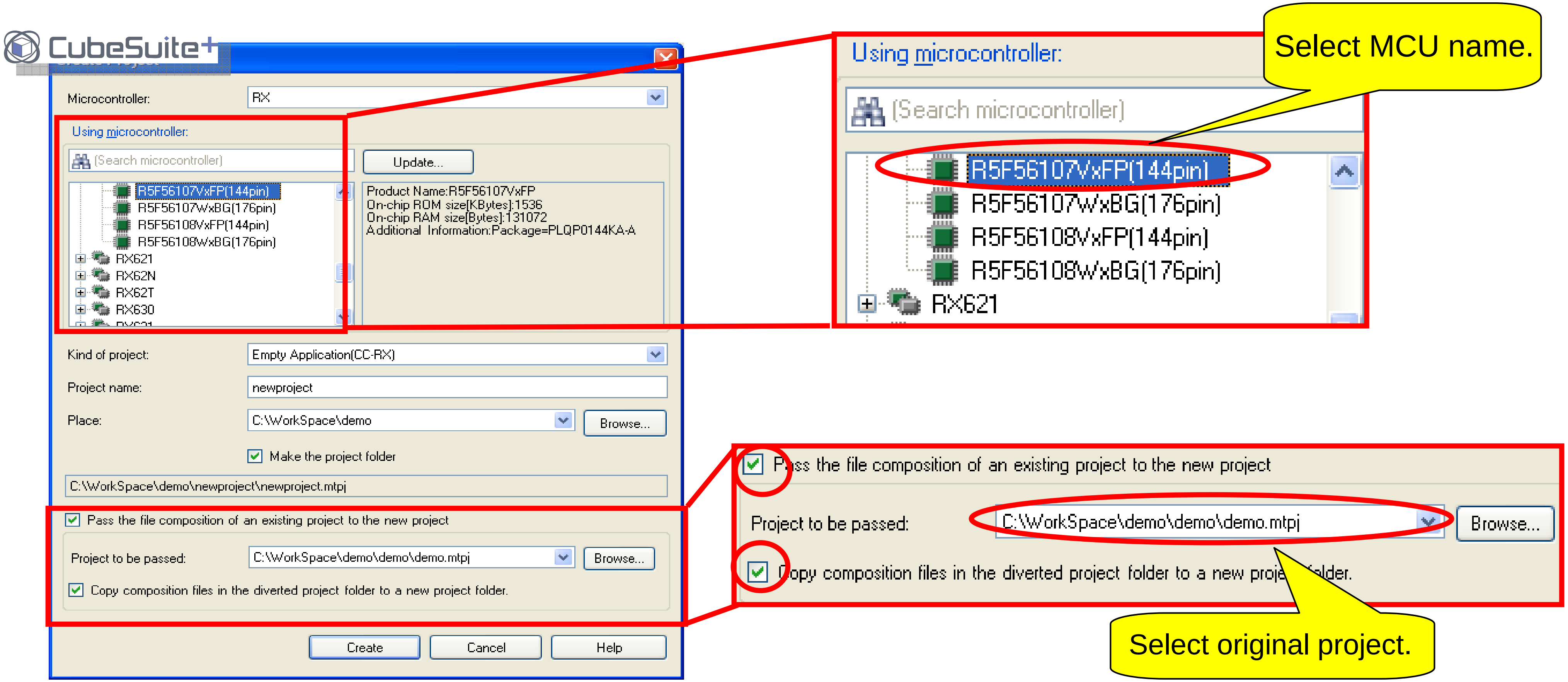
The new project can be created from the CubeSuite, High-performance Embedded Workshop and PM+ projects by opening.
Supported High-performance Embedded Workshop project version is 4.07.00 and later for RX. Supported PM+ project version is 6.00 and later for V850/78K0R/78K0. The new projects are also created for each project group.
The created project files are saved to the original project folder.

```
[Direct Error Cause]
The conversion of project demo was aborted(E0292009).
[EOF]
```

3. Changing the MCU

(3) The [Create Project] dialog box opens. Select the name of the MCU you want to use in [Using microcontroller].

(4) Select the [Pass the file composition of an existing project to the new project] checkbox, select the original project, and then select the［Copy composition files in the diverted project folder to a new project folder］checkbox.



Microcontroller: RX

Using microcontroller:

(Search microcontroller)

- R5F56107VxFP(144pin)
- R5F56107WxBG(176pin)
- R5F56108VxFP(144pin)
- R5F56108WxBG(176pin)

Product Name:R5F56107VxFP
On-chip ROM size[KBytes]:1536
On-chip RAM size[Bytes]:131072
Additional Information:Package=PLQP0144KA-A

Kind of project: Empty Application(CC-RX)

Project name: newproject

Place: C:\WorkSpace\demo

☑ Make the project folder

C:\WorkSpace\demo\newproject\newproject.mtpj

☑ Pass the file composition of an existing project to the new project

Project to be passed: C:\WorkSpace\demo\demo.mtpj

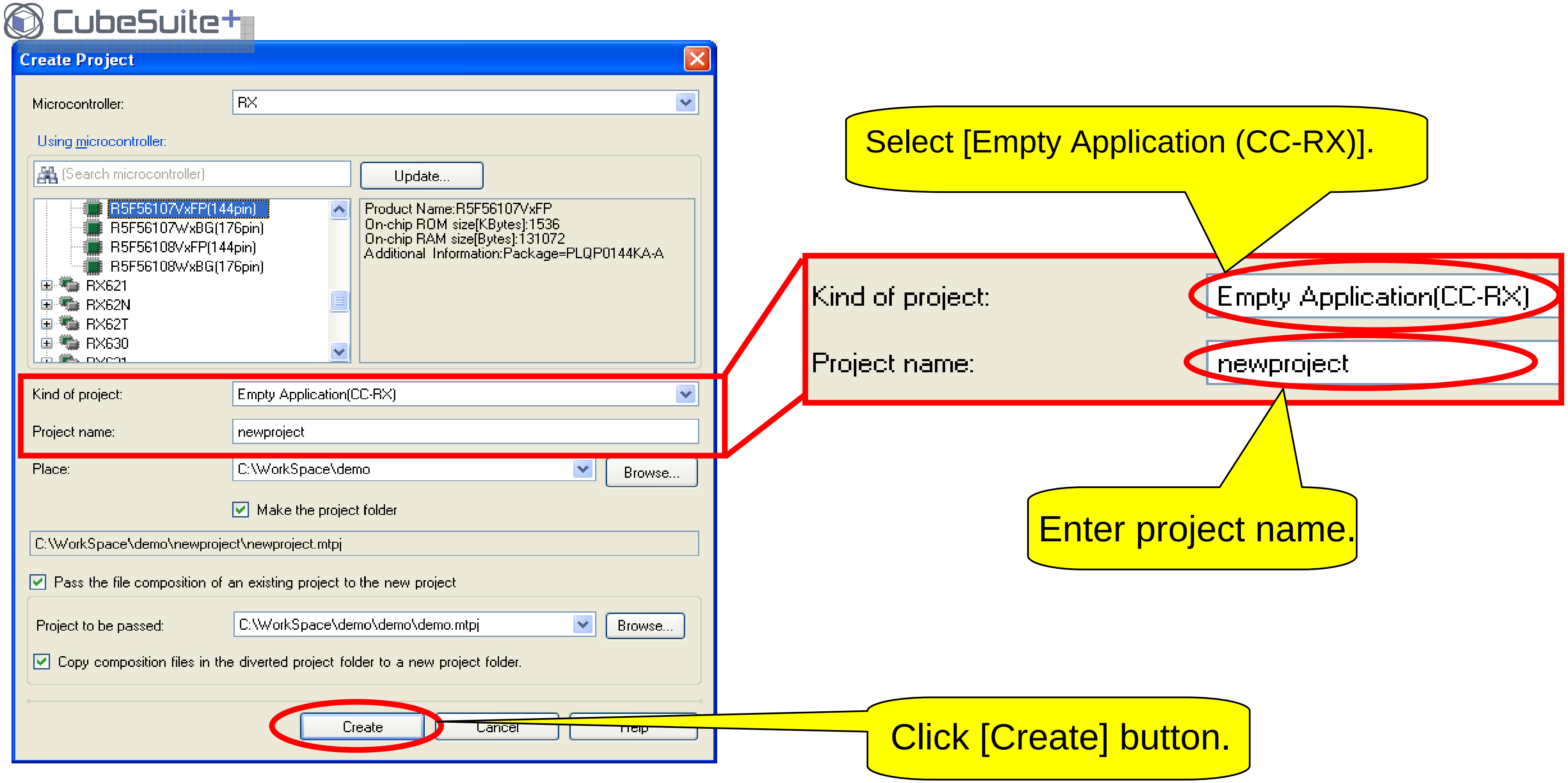
☑ Copy composition files in the diverted project folder to a new project folder.

Select MCU name.

Select original project.

3. Changing the MCU

(5) Specify [Kind of project] and [Project name] and click the [Create] button. The newly created project is based on the original project but the MCU has been changed.



Create Project

Microcontroller: RX

Using microcontroller:

Product Name:R5F56107VxFP
On-chip ROM size[KBytes]:1536
On-chip RAM size[Bytes]:131072
Additional Information:Package=PLQP0144KA-A

Kind of project: Empty Application(CC-RX)

Project name: newproject

Place: C:\WorkSpace\demo

☑ Make the project folder

C:\WorkSpace\demo\newproject\newproject.mtpj

☑ Pass the file composition of an existing project to the new project

Project to be passed: C:\WorkSpace\demo\demo\demo.mtpj

☑ Copy composition files in the diverted project folder to a new project folder.

Select [Empty Application (CC-RX)].

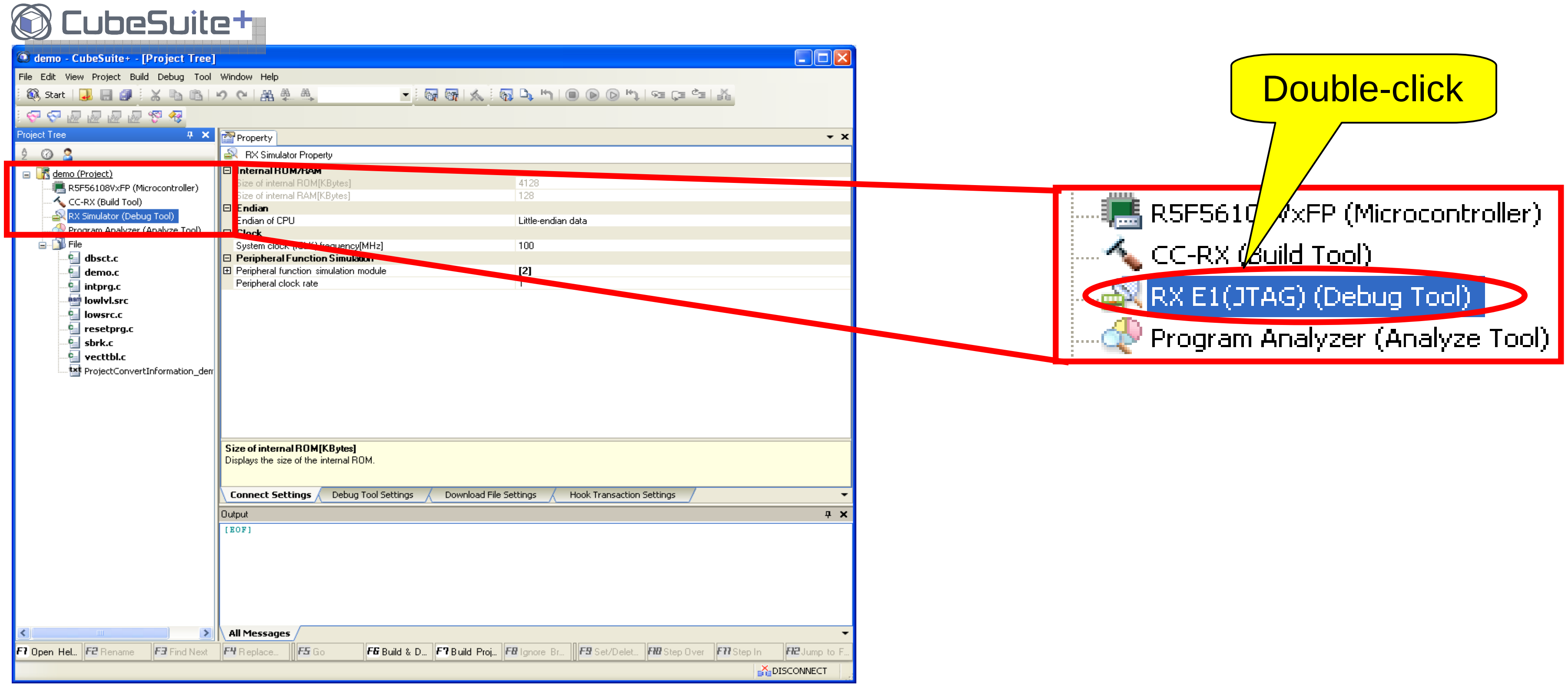
Enter project name.

Click [Create] button.

4. Where Do We Make Settings when Connecting an Emulator?

In the HEW, the [Initial Settings] and [Configuration Properties] dialog boxes open to make settings when connecting an emulator. In the CubeSuite+, on the other hand, you need to make settings on the [Properties] panel before connecting an emulator by taking the following procedure.

Double-click the debug tool name (debug tool) on the [Project Tree] panel to open the Properties window of the debug tool.



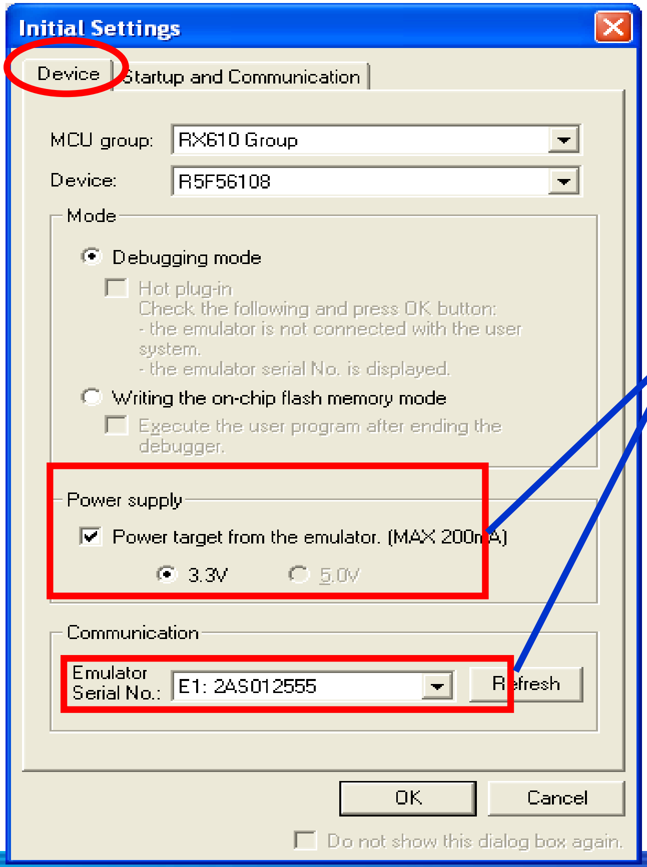
Internal ROM/RAM	
Size of internal ROM[KBytes]	4128
Size of internal RAM[KBytes]	128
Endian	
Endian of CPU	Little-endian data
Clock	
System clock (ICLK) frequency[MHz]	100
Peripheral Function Simulation	
Peripheral function simulation module	[2]
Peripheral clock rate	1

Size of internal ROM[KBytes]
Displays the size of the internal ROM.

4. Where Do We Make Settings when Connecting an Emulator?

The following figures show the correspondence between settings on the HEW‘s [Initial Settings] and [Configuration Properties] dialog boxes and settings on the [Properties] panel of the CubeSuite+.

(1) [Device] page of the [Initial Settings] dialog box in the HEW



Initial Settings

Device | Startup and Communication

MCU group: RX610 Group

Device: R5F56108

Mode

● Debugging mode

□ Hot plug-in
Check the following and press OK button:
- the emulator is not connected with the user system.
- the emulator serial No. is displayed.

○ Writing the on-chip flash memory mode

□ Execute the user program after ending the debugger.

Power supply

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

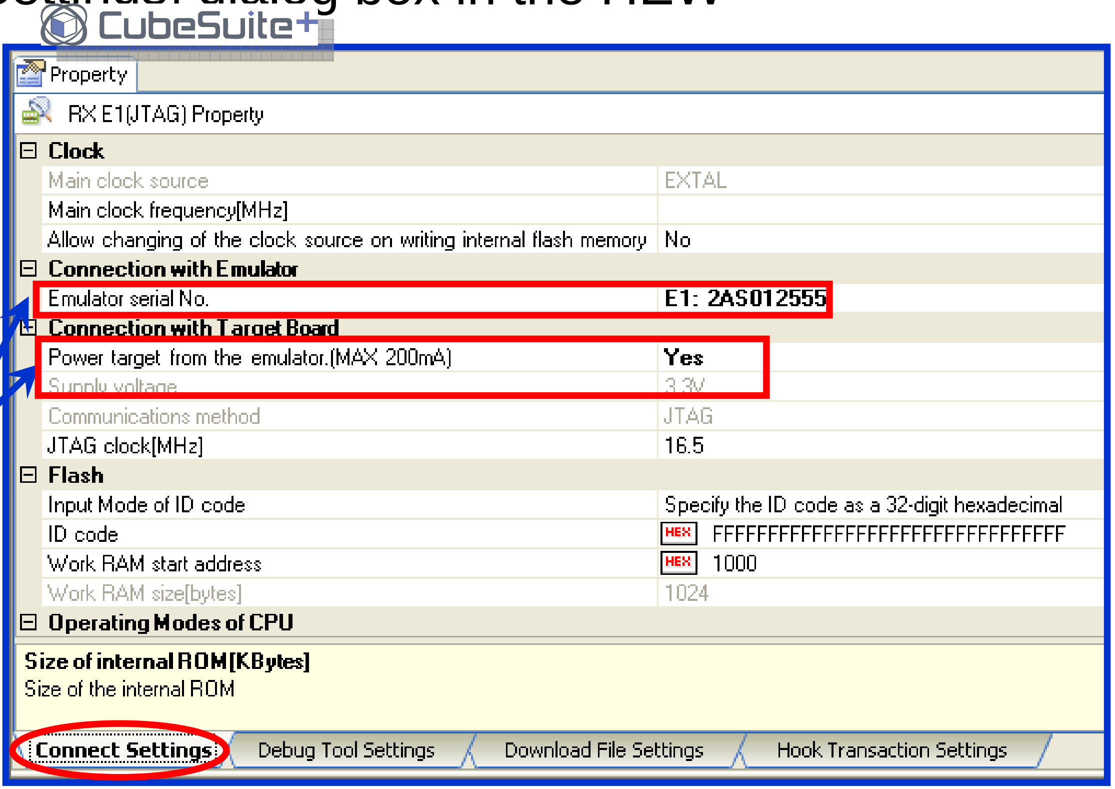
● 3.3V ○ 5.0V

Communication

Emulator Serial No.: E1: 2AS012555 [Refresh]

[OK] [Cancel]

□ Do not show this dialog box again.



CubeSuite+

Property

RX E1(JTAG) Property

⊟ **Clock**	
Main clock source	EXTAL
Main clock frequency[MHz]	
Allow changing of the clock source on writing internal flash memory	No
⊟ **Connection with Emulator**	
Emulator serial No.	**E1: 2AS012555**
⊟ **Connection with Target Board**	
Power target from the emulator.(MAX 200mA)	**Yes**
Supply voltage	3.3V
Communications method	JTAG
JTAG clock[MHz]	16.5
⊟ **Flash**	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024
⊟ **Operating Modes of CPU**	

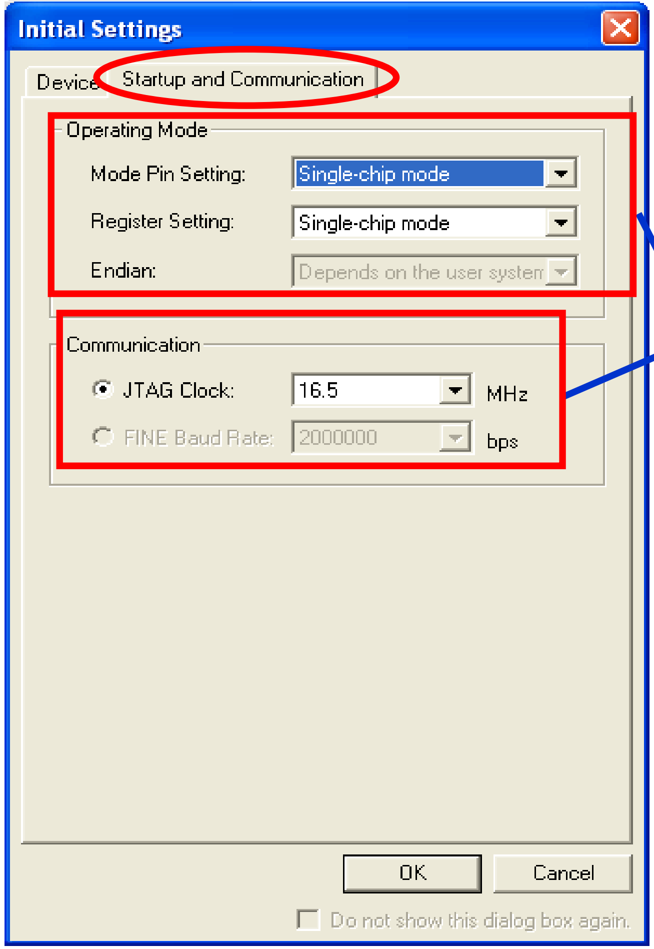
Size of internal ROM[KBytes]
Size of the internal ROM

Note: It is not possible to change the device on the CubeSuite+.
For how to change the device, see “3. Changing the MCU.”

4. Where Do We Make Settings when Connecting an Emulator?

(2) [Startup and Communication] page of the [Initial Settings] dialog box in the HEW

High-performance Embedded Workshop



Initial Settings

Device | Startup and Communication

Operating Mode

Mode Pin Setting: Single-chip mode

Register Setting: Single-chip mode

Endian: Depends on the user systen

Communication

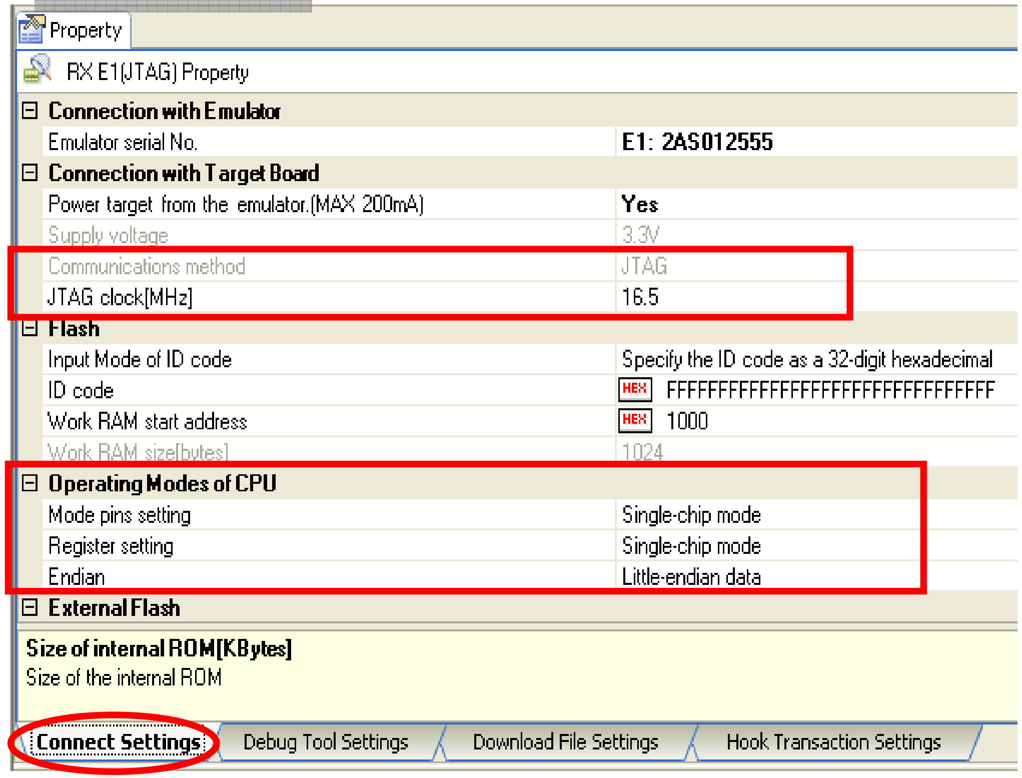
● JTAG Clock: 16.5 MHz

○ FINE Baud Rate: 2000000 bps

OK | Cancel

☐ Do not show this dialog box again.

CubeSuite+



Property

RX E1(JTAG) Property

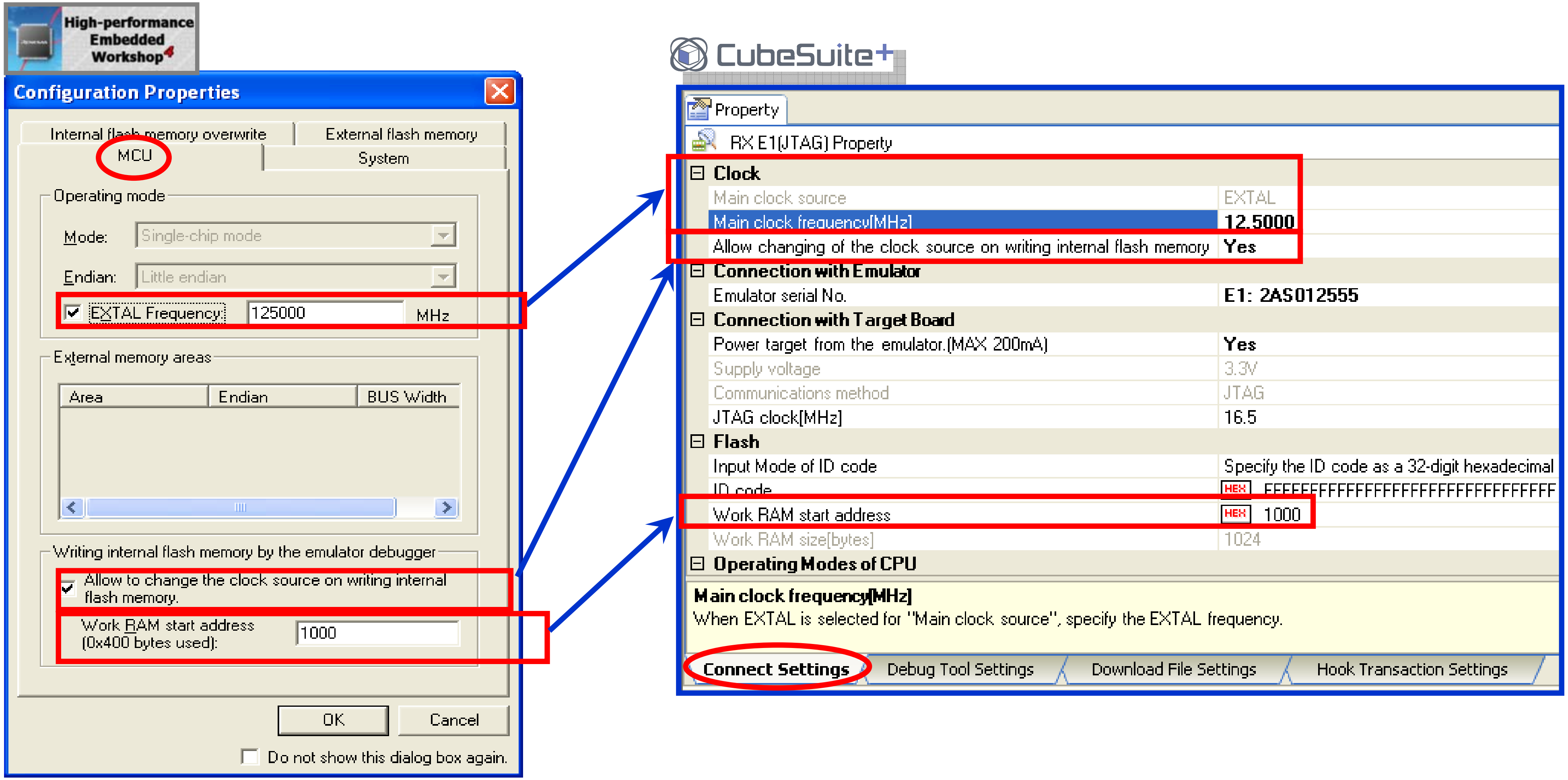
⊟ **Connection with Emulator**	
Emulator serial No.	E1: 2AS012555
⊟ **Connection with Target Board**	
Power target from the emulator.(MAX 200mA)	Yes
Supply voltage	3.3V
Communications method	JTAG
JTAG clock[MHz]	16.5
⊟ **Flash**	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024
⊟ **Operating Modes of CPU**	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data
⊟ **External Flash**	

Size of internal ROM[KBytes]
Size of the internal ROM

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

4. Where Do We Make Settings when Connecting an Emulator?

(3) [MCU] page of the [Configuration Properties] dialog box in the HEW


High-performance Embedded Workshop4
CubeSuite+

Configuration Properties

Internal flash memory overwrite
External flash memory
MCU
System

Operating mode

Mode: Single-chip mode

Endian: Little endian

☑ EXTAL Frequency: 125000 MHz

External memory areas

Area	Endian	BUS Width

Writing internal flash memory by the emulator debugger

☑ Allow to change the clock source on writing internal flash memory.

Work RAM start address (0x400 bytes used): 1000

□ Do not show this dialog box again.

RX E1(JTAG) Property

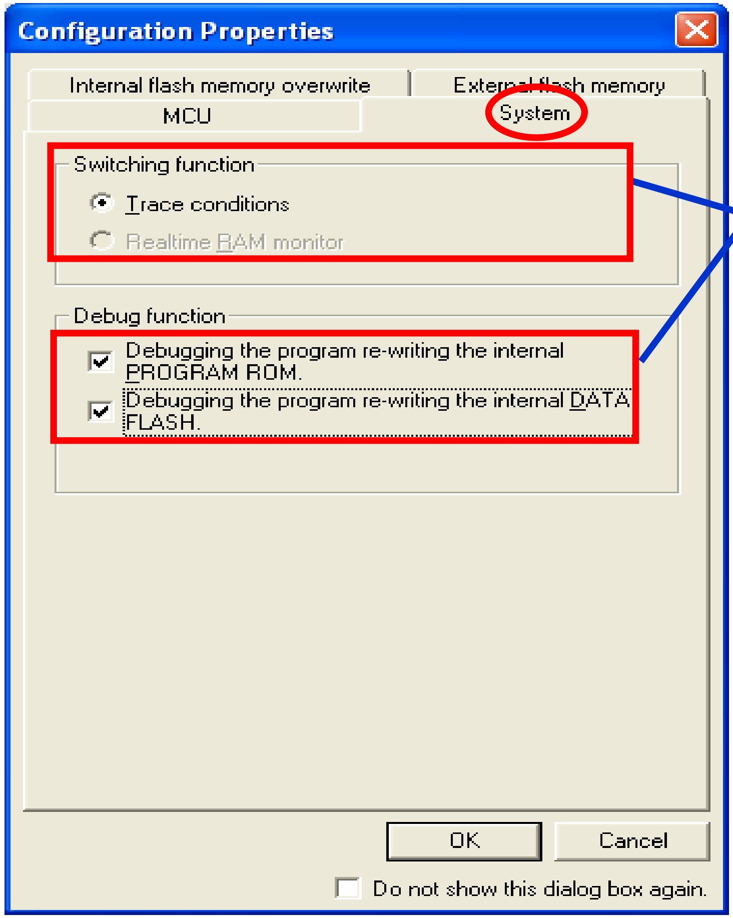
⊟ **Clock**	
Main clock source	EXTAL
Main clock frequency[MHz]	12.5000
Allow changing of the clock source on writing internal flash memory	Yes
⊟ **Connection with Emulator**	
Emulator serial No.	E1: 2AS012555
⊟ **Connection with Target Board**	
Power target from the emulator.(MAX 200mA)	Yes
Supply voltage	3.3V
Communications method	JTAG
JTAG clock[MHz]	16.5
⊟ **Flash**	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024
⊟ **Operating Modes of CPU**	

Main clock frequency[MHz]
When EXTAL is selected for "Main clock source", specify the EXTAL frequency.

4. Where Do We Make Settings when Connecting an Emulator?

(4) [System] page of the [Configuration Properties] dialog box in the HEW

High-performance Embedded Workshop



Configuration Properties

Internal flash memory overwrite | External flash memory | MCU | System

Switching function

● Trace conditions

○ Realtime RAM monitor

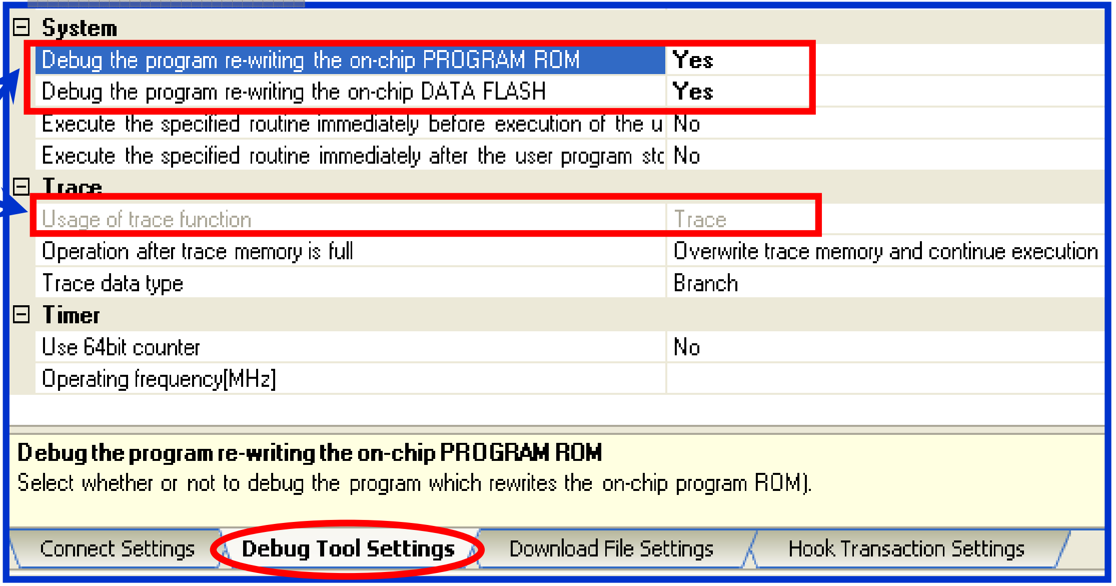
Debug function

☑ Debugging the program re-writing the internal PROGRAM ROM.

☑ Debugging the program re-writing the internal DATA FLASH.

□ Do not show this dialog box again.

CubeSuite+



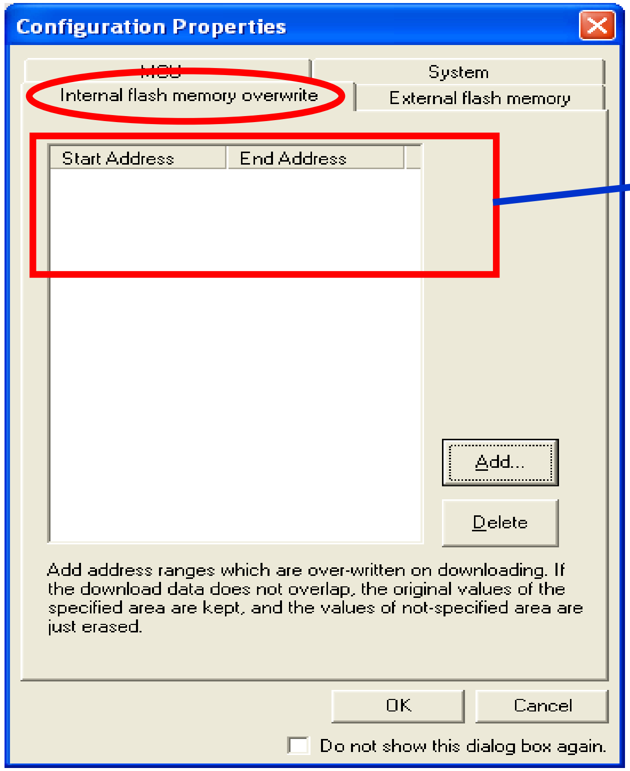
System	
Debug the program re-writing the on-chip PROGRAM ROM	Yes
Debug the program re-writing the on-chip DATA FLASH	Yes
Execute the specified routine immediately before execution of the u	No
Execute the specified routine immediately after the user program stc	No
Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch
Timer	
Use 64bit counter	No
Operating frequency[MHz]	

Debug the program re-writing the on-chip PROGRAM ROM
Select whether or not to debug the program which rewrites the on-chip program ROM).

4. Where Do We Make Settings when Connecting an Emulator?

(5) [Internal flash memory overwrite] page of the [Configuration Properties] dialog box in the HEW

High-performance Embedded Workshop4



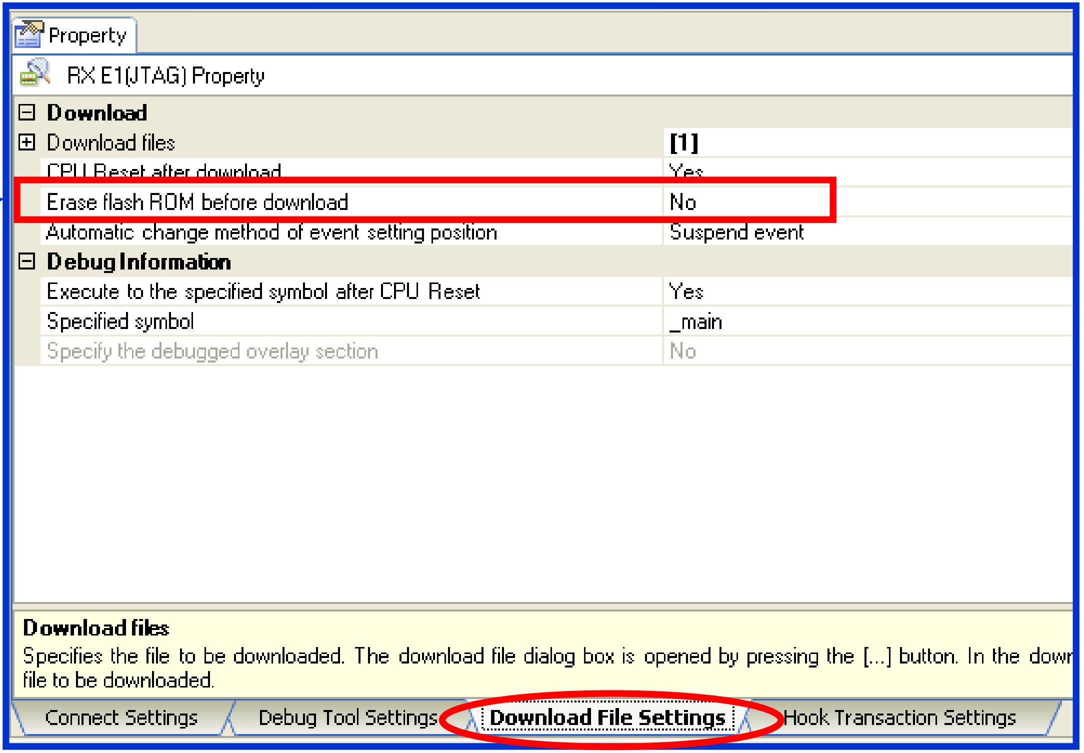
Configuration Properties

Start Address	End Address

Add address ranges which are over-written on downloading. If the download data does not overlap, the original values of the specified area are kept, and the values of not-specified area are just erased.

□ Do not show this dialog box again.

CubeSuite+



RX E1(JTAG) Property

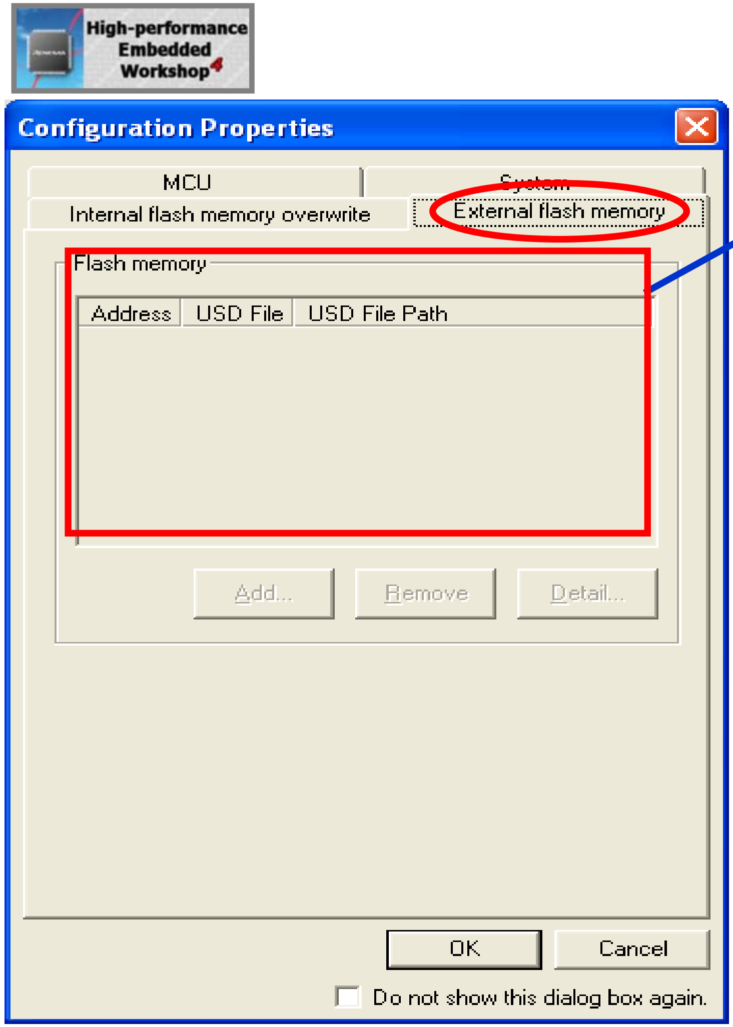
⊟ **Download**	
⊞ Download files	**[1]**
CPU Reset after download	Yes
Erase flash ROM before download	No
Automatic change method of event setting position	Suspend event
⊟ **Debug Information**	
Execute to the specified symbol after CPU Reset	Yes
Specified symbol	_main
Specify the debugged overlay section	No

Download files
Specifies the file to be downloaded. The download file dialog box is opened by pressing the [...] button. In the dowr file to be downloaded.

Note: The erasing and overwriting settings are made for the entire flash area, and cannot be made for each block.

4. Where Do We Make Settings when Connecting an Emulator?

(6) [External flash memory] page of the [Configuration Properties] dialog box in the HEW



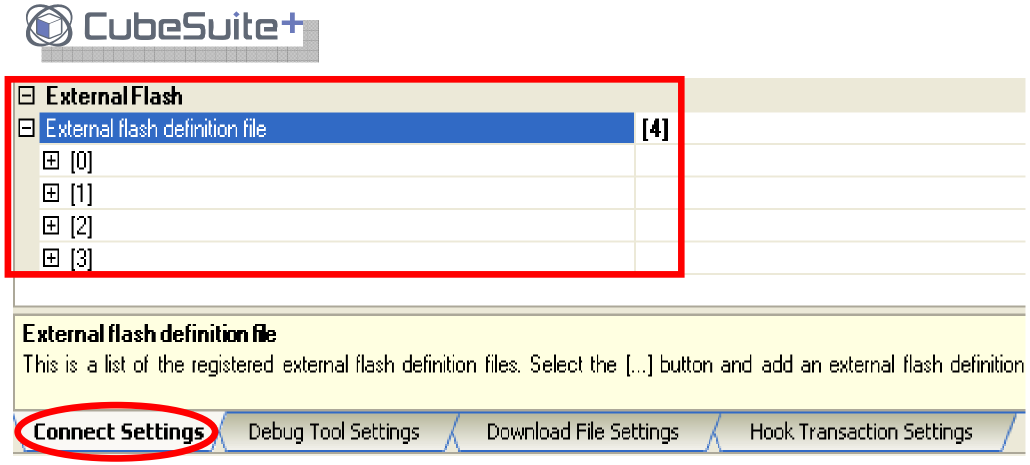
Configuration Properties

MCU | System | Internal flash memory overwrite | External flash memory

Flash memory

Address	USD File	USD File Path

□ Do not show this dialog box again.



⊟ External Flash

⊟ External flash definition file [4]

⊞ [0]

⊞ [1]

⊞ [2]

⊞ [3]

External flash definition file

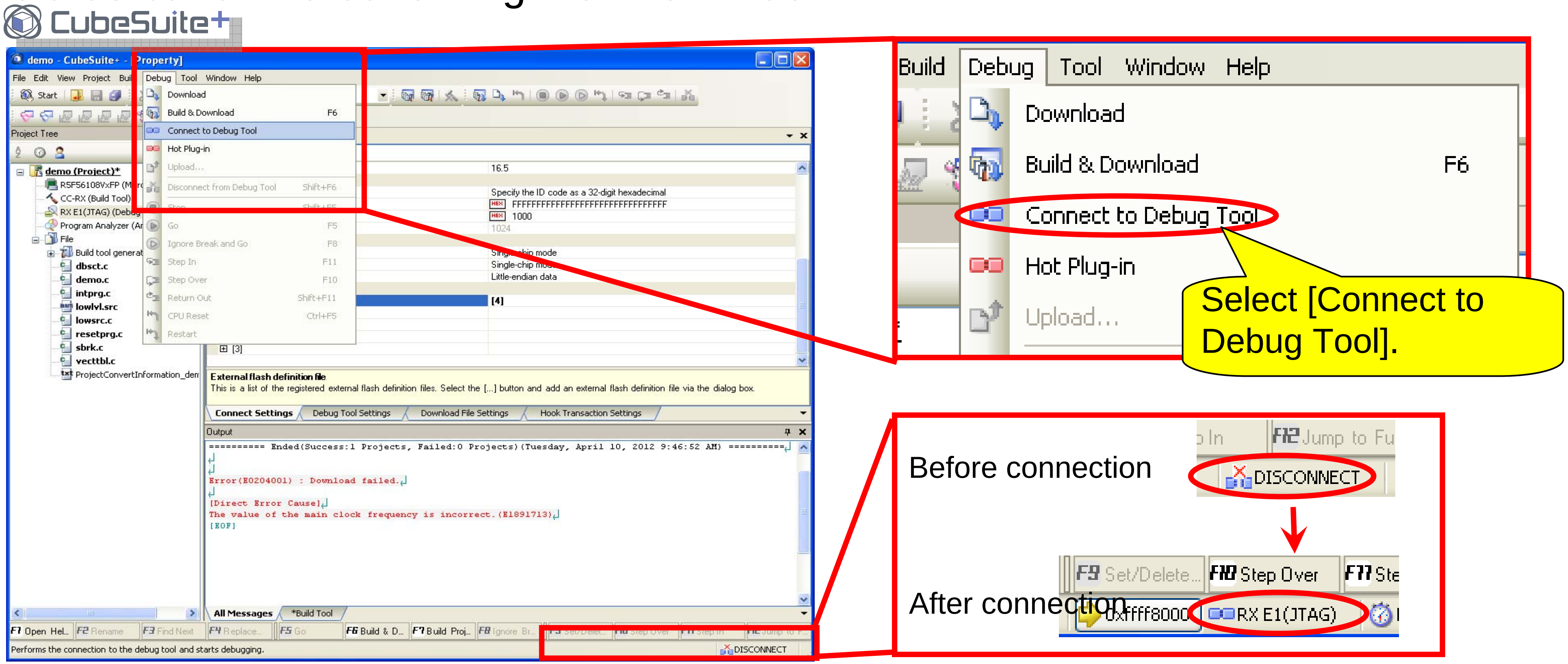
This is a list of the registered external flash definition files. Select the [...] button and add an external flash definition

Note: USD files used in the HEW can be registered as as they are.

They do not need to be modified even if a pre-download execution script and a post-download execution script have been registered.

5. Connecting an Emulator

Select [Debug] -> [Connect to Debug Tool] from the CubeSuite+ menu to establish connection to the selected emulator (debug tool).
Upon completion of the connection, the debug tool name appears on the status bar at the bottom right of the window.



External flash definition file
This is a list of the registered external flash definition files. Select the [...] button and add an external flash definition file via the dialog box.

```
========== Ended(Success:1 Projects, Failed:0 Projects)(Tuesday, April 10, 2012 9:46:52 AM) ==========


Error(E0204001) : Download failed.

[Direct Error Cause]
The value of the main clock frequency is incorrect.(E1891713)
[EOF]
```

Performs the connection to the debug tool and starts debugging.

Select [Connect to Debug Tool].

Before connection

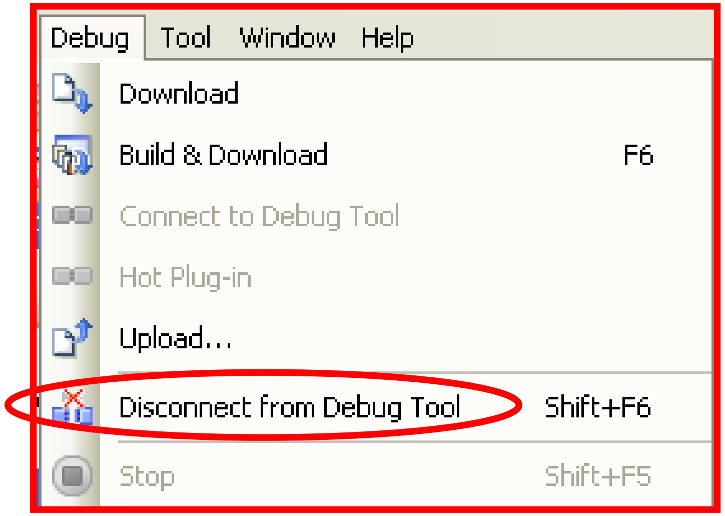
After connection

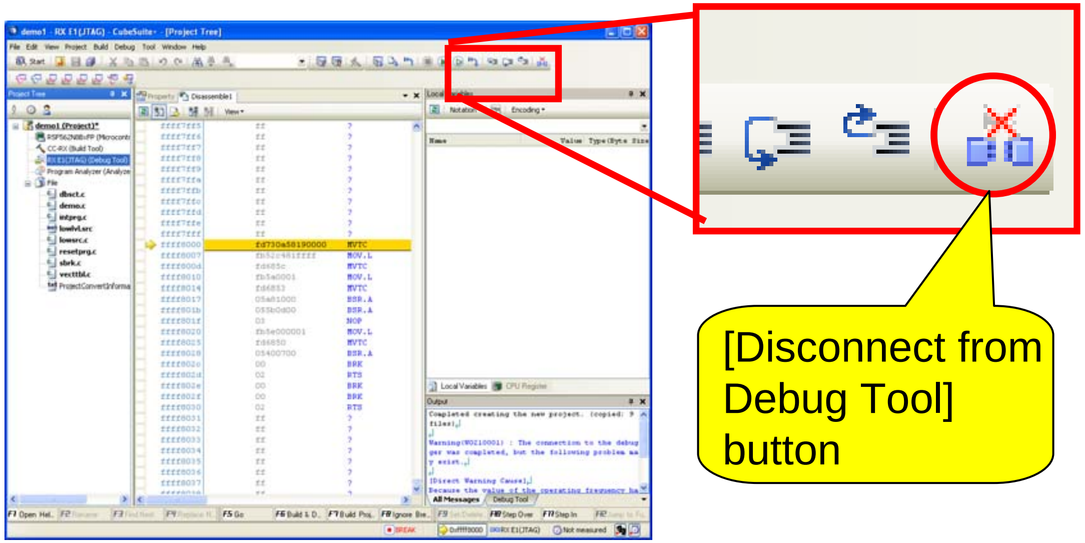
Note: If an ID code has been written in the MCU, set an ID code in advance according to “8. Entering an ID Code.”

6. Disconnecting the Emulator

To disconnect the emulator, select [Disconnect from Debug Tool] from the menu or click the button on the debug toolbar.

CubeSuite+



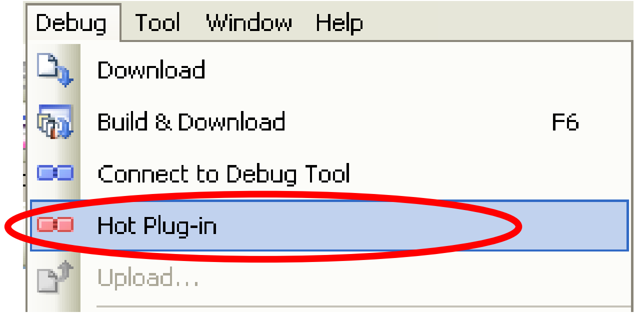


[Disconnect from Debug Tool] button

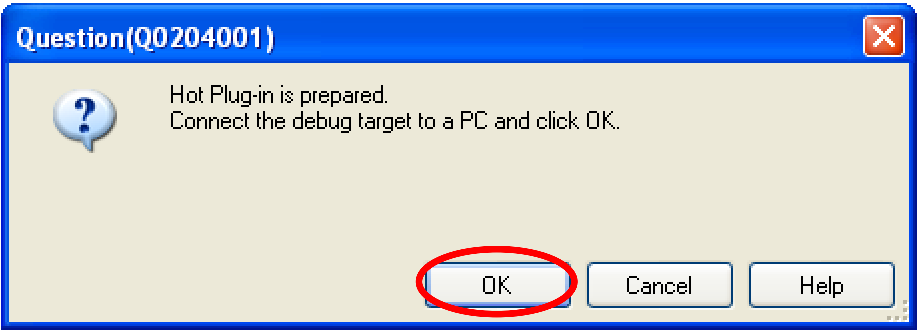
7. Starting an Emulator with Hot Plug-In

The following describes a procedure to connect an emulator by using the hot plug-in facility.

(1) Select [Debug] -> [Hot Plug-in] from the CubeSuite+ menu.



(2) The following message appears. Connect the emulator to the target board and then click the [OK] button to start up the emulator.

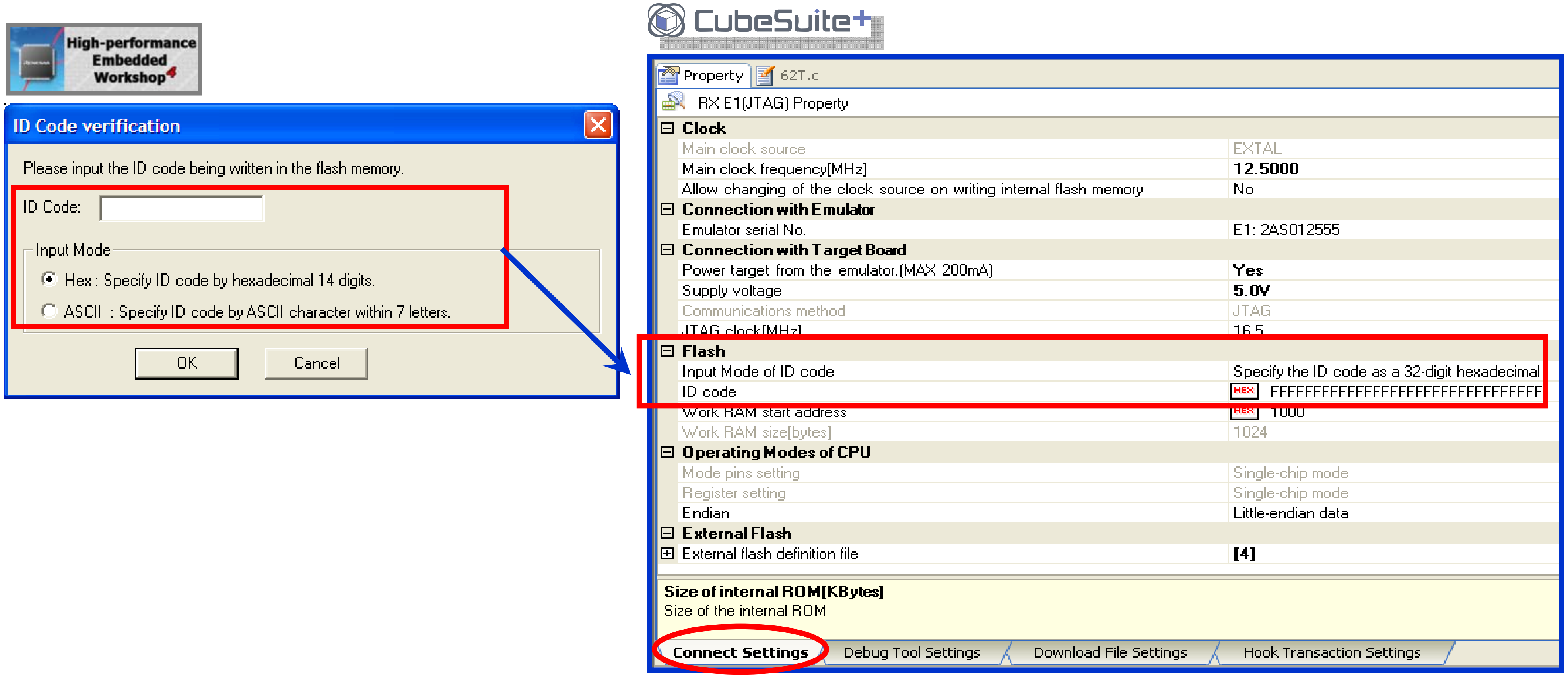

Question(Q0204001)

Hot Plug-in is prepared.
Connect the debug target to a PC and click OK.

Note: If an ID code has been written in the MCU, set an ID code in advance according to “8. Entering an ID Code.”

8. Entering an ID Code

In the HEW, the [ID Code verification] dialog box opens at startup if an ID code has been written in the MCU. In CubeSuite+, on the other hand, an ID code must be set on the [Properties] panel before the emulator is started up.
Set an ID code by referring to the following figures:



High-performance Embedded Workshop

ID Code verification

Please input the ID code being written in the flash memory.

ID Code:

Input Mode

● Hex : Specify ID code by hexadecimal 14 digits.

○ ASCII : Specify ID code by ASCII character within 7 letters.

RX E1(JTAG) Property

Clock	
Main clock source	EXTAL
Main clock frequency[MHz]	**12.5000**
Allow changing of the clock source on writing internal flash memory	No
Connection with Emulator	
Emulator serial No.	E1: 2AS012555
Connection with Target Board	
Power target from the emulator.(MAX 200mA)	**Yes**
Supply voltage	**5.0V**
Communications method	JTAG
JTAG clock[MHz]	16.5
Flash	
Input Mode of ID code	Specify the ID code as a 32-digit hexadecimal
ID code	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Work RAM start address	HEX 1000
Work RAM size[bytes]	1024
Operating Modes of CPU	
Mode pins setting	Single-chip mode
Register setting	Single-chip mode
Endian	Little-endian data
External Flash	
External flash definition file	**[4]**

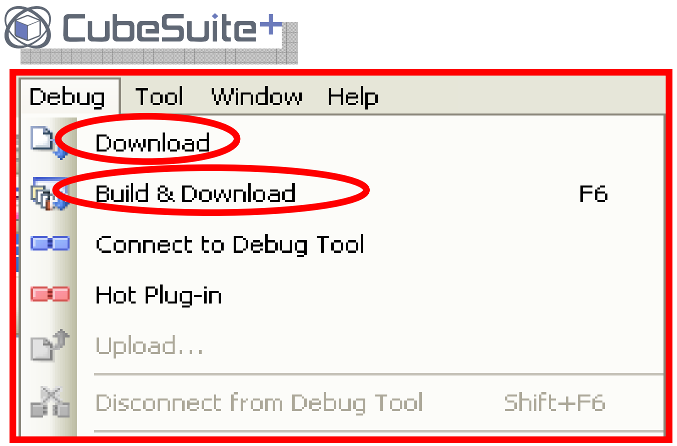
Size of internal ROM[KBytes]
Size of the internal ROM

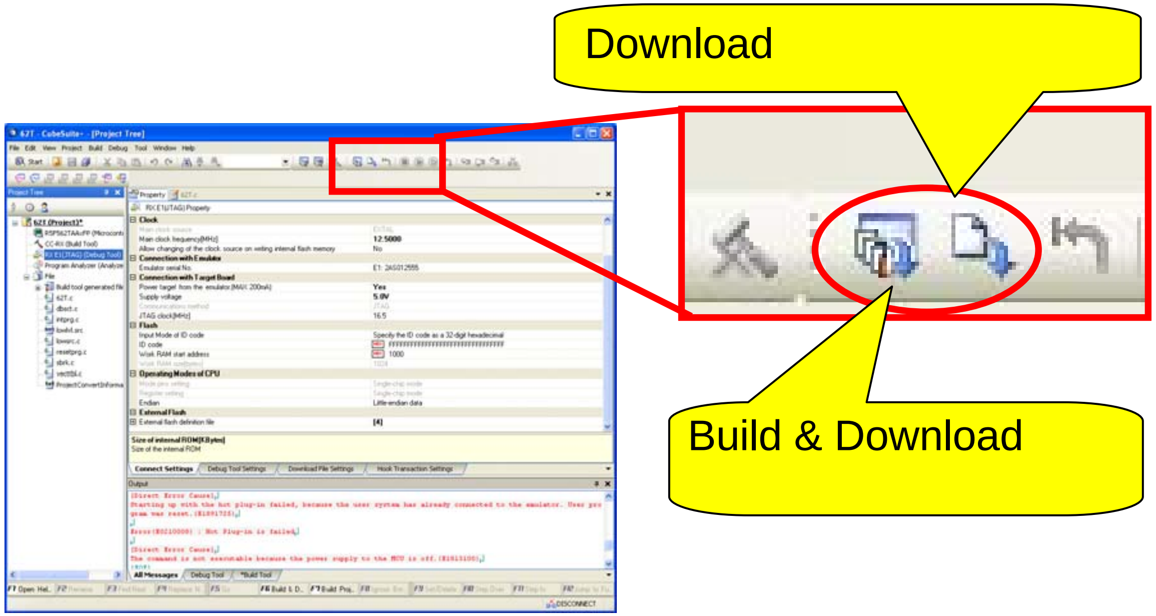
9. Downloading a Program

Selecting [Debug] -> [Download] from the menu or clicking the button on the debug toolbar starts downloading specified files.

Selecting [Debug] -> [Build & Download] from the menu or clicking the button on the debug toolbar builds a project and then starts downloading the specified files.

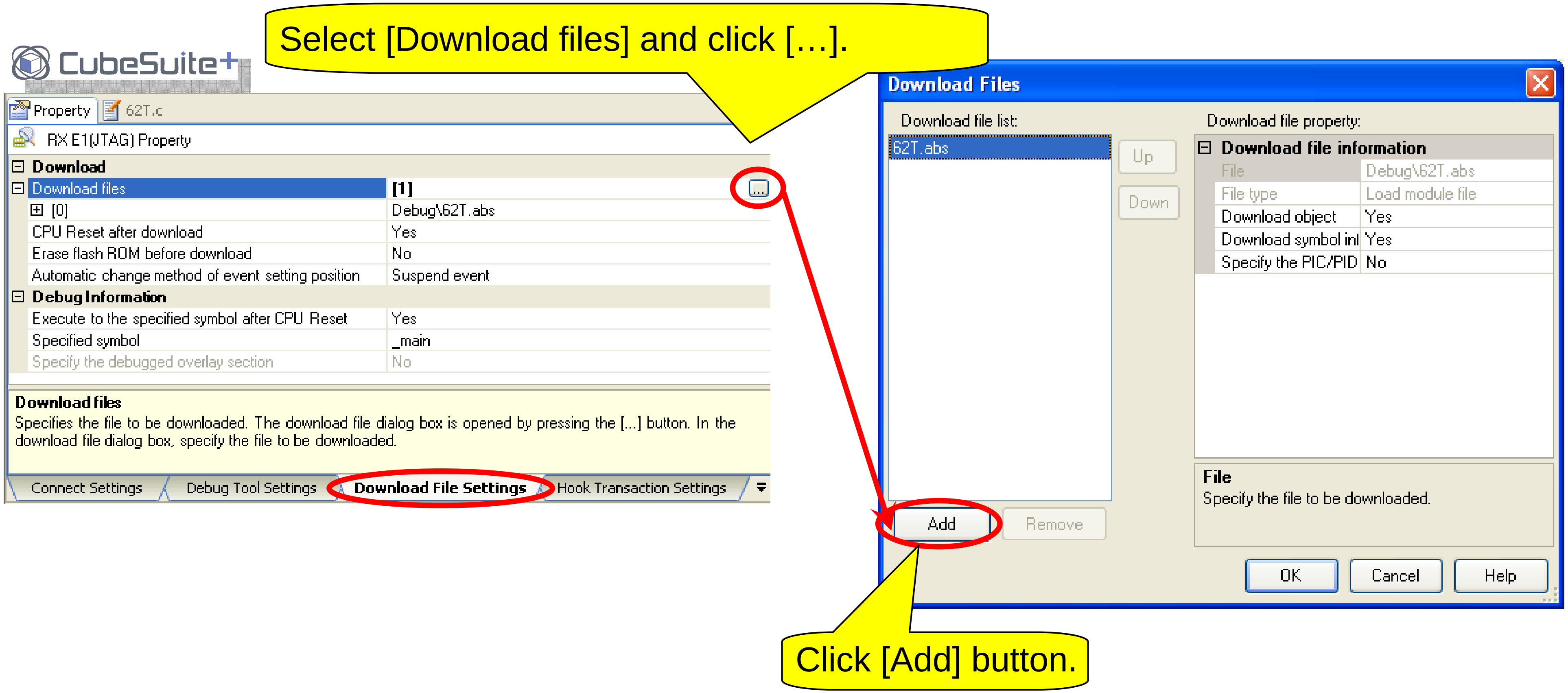
If no debug tool is connected, connection to a debug tool must be established before downloading.





10. Registering Additional Download Files

Add download files in the [Download File Settings] sheet on the [Properties] panel.
(1) Select [Download files] and click […] on the right.
(2) The [Download Files] dialog box opens. Click the [Add] button.



Select [Download files] and click […].

CubeSuite+

Property | 62T.c

RX E1(JTAG) Property

⊟ **Download**	
⊟ Download files	[1]
⊞ [0]	Debug\62T.abs
CPU Reset after download	Yes
Erase flash ROM before download	No
Automatic change method of event setting position	Suspend event
⊟ **Debug Information**	
Execute to the specified symbol after CPU Reset	Yes
Specified symbol	_main
Specify the debugged overlay section	No

Download files
Specifies the file to be downloaded. The download file dialog box is opened by pressing the [...] button. In the download file dialog box, specify the file to be downloaded.

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

Download Files

Download file list:

62T.abs

Download file property:

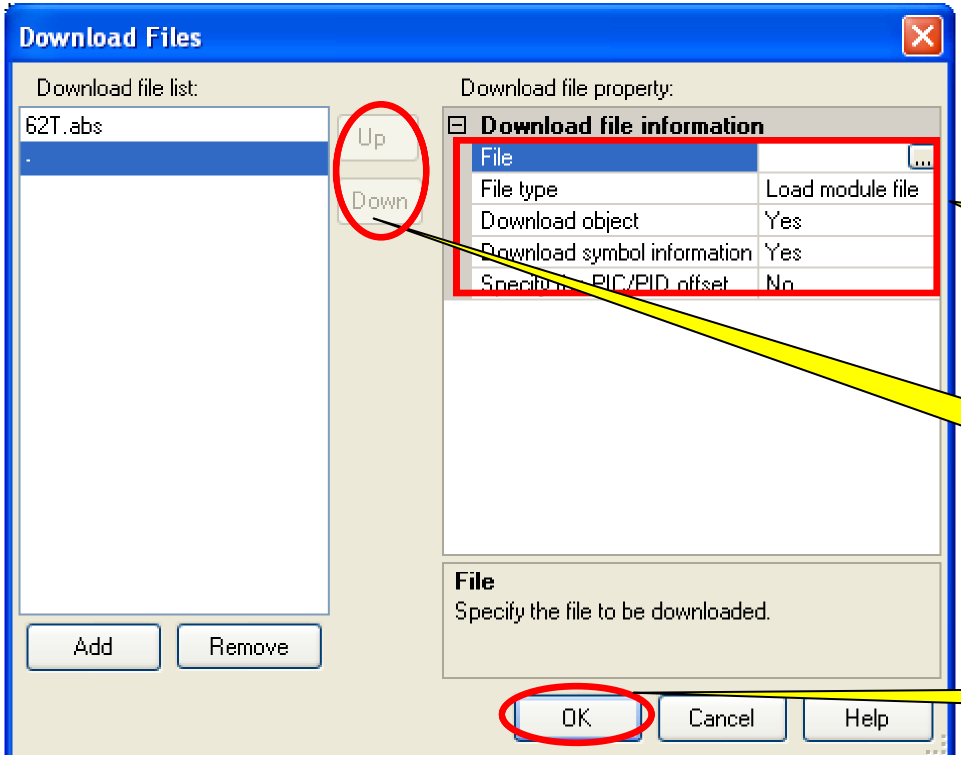
⊟ **Download file information**	
File	Debug\62T.abs
File type	Load module file
Download object	Yes
Download symbol inf	Yes
Specify the PIC/PID	No

File
Specify the file to be downloaded.

Click [Add] button.

10. Registering Additional Download Files

(3) Specify the file name and file type in the [Download file information] field and then click the [OK] button.



Download Files

Download file list:

- 62T.abs
- -

Download file property:

⊟ Download file information

File	...
File type	Load module file
Download object	Yes
Download symbol information	Yes
Specify the PIC/PID offset	No

File
Specify the file to be downloaded.

Specify download file and file type.

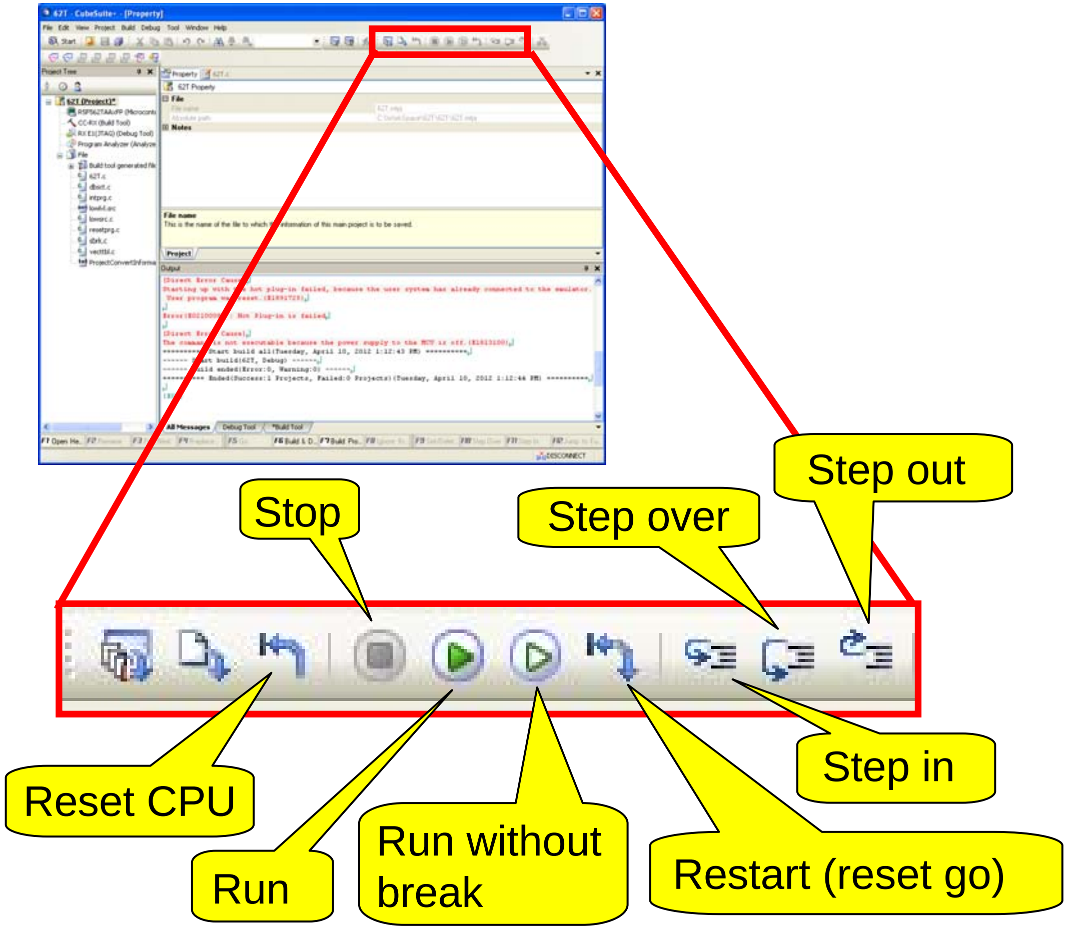
Use these buttons to select download order.

Click [OK] button.

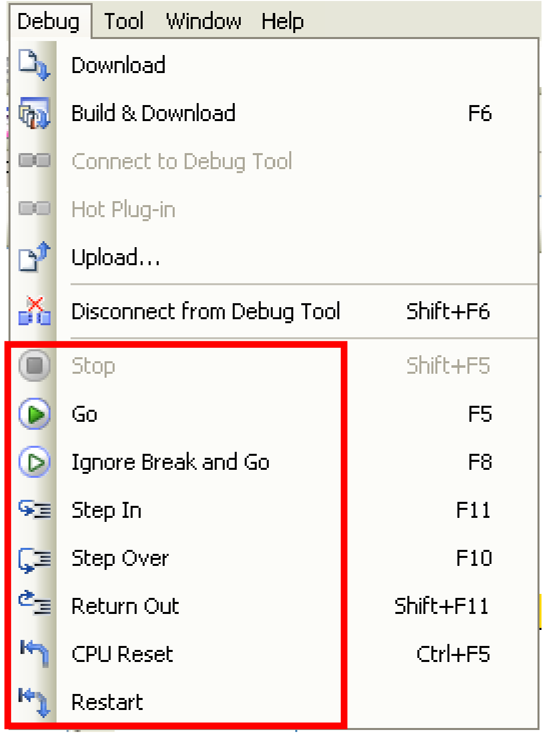
Note: When downloading is performed, all of the registered files are downloaded. To download only desired files, set [Download object] and [Download symbol information] to “Yes” in this window only for files you want to download.

11. Starting/Stopping a Program

You can start or stop a program and reset the CPU from the menu or toolbar in the same way as the HEW (see below).

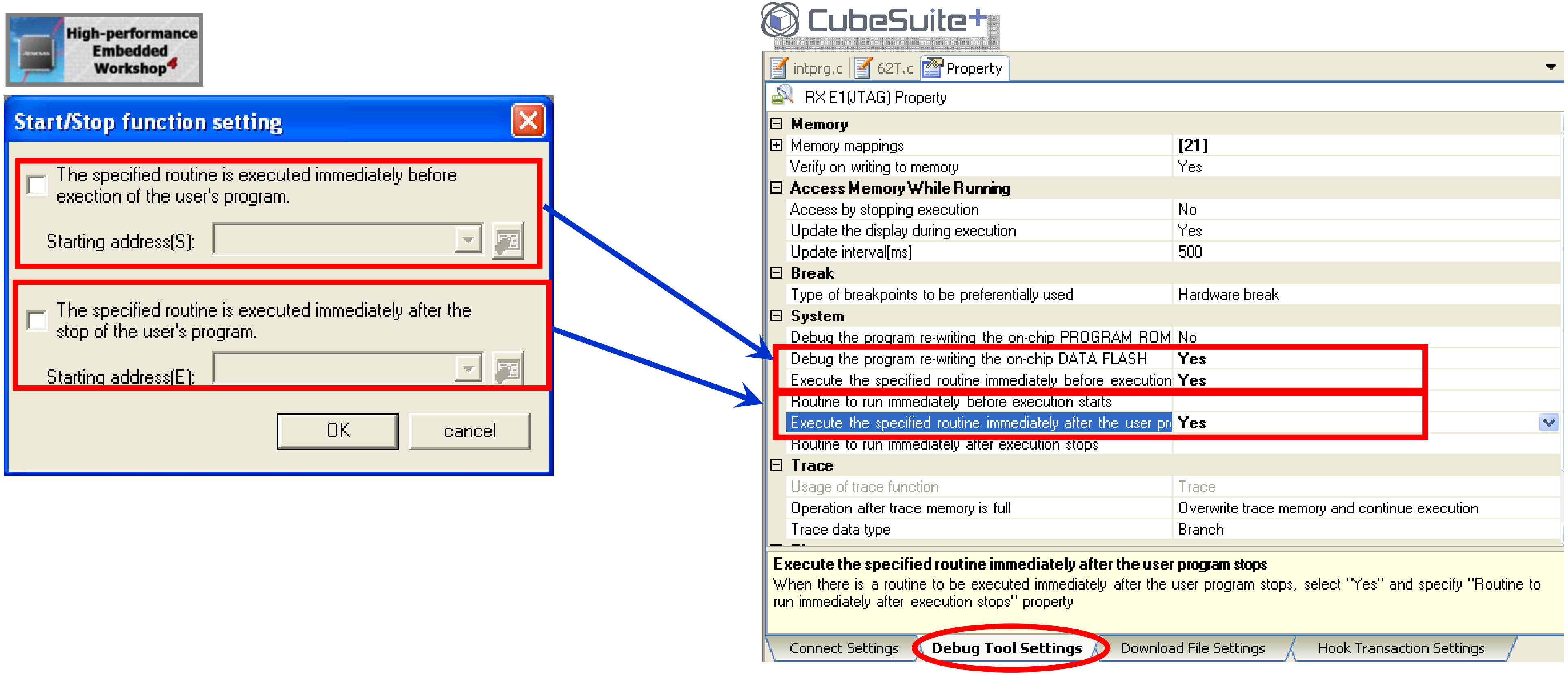


CubeSuite+ menu



12. Setting the Start/Stop Function

You can set the Start/Stop function in the [Debug Tool Settings] sheet on the [Properties] panel (see below).



Start/Stop function setting

☐ The specified routine is executed immediately before exection of the user's program.

Starting address(S):

☐ The specified routine is executed immediately after the stop of the user's program.

Starting address(E):

RX E1(JTAG) Property

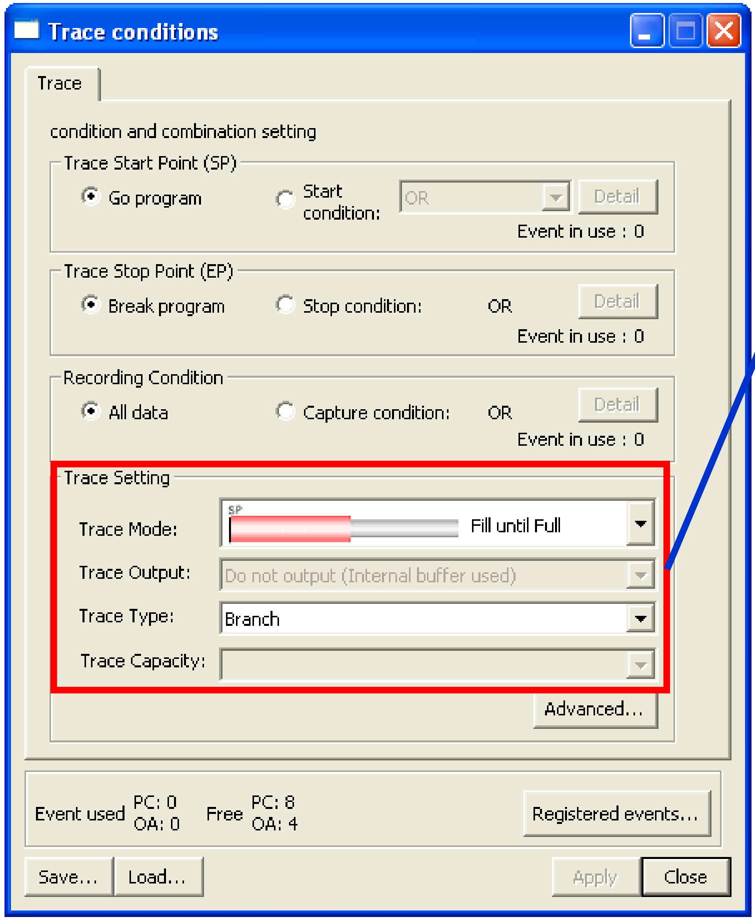
⊟ **Memory**	
⊞ Memory mappings	**[21]**
Verify on writing to memory	Yes
⊟ **Access Memory While Running**	
Access by stopping execution	No
Update the display during execution	Yes
Update interval[ms]	500
⊟ **Break**	
Type of breakpoints to be preferentially used	Hardware break
⊟ **System**	
Debug the program re-writing the on-chip PROGRAM ROM	No
Debug the program re-writing the on-chip DATA FLASH	**Yes**
Execute the specified routine immediately before execution	**Yes**
Routine to run immediately before execution starts	
Execute the specified routine immediately after the user pr	**Yes**
Routine to run immediately after execution stops	
⊟ **Trace**	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch

Execute the specified routine immediately after the user program stops
When there is a routine to be executed immediately after the user program stops, select "Yes" and specify "Routine to run immediately after execution stops" property

13. Setting Trace Acquisition Conditions

You can set trace acquisition conditions in the [Debug Tool Settings] sheet on the [Properties] panel (see below).

High-performance Embedded Workshop



Trace conditions

Trace

condition and combination setting

Trace Start Point (SP)

● Go program ○ Start condition: OR Detail

Event in use : 0

Trace Stop Point (EP)

● Break program ○ Stop condition: OR Detail

Event in use : 0

Recording Condition

● All data ○ Capture condition: OR Detail

Event in use : 0

Trace Setting

Trace Mode: Fill until Full

Trace Output: Do not output (Internal buffer used)

Trace Type: Branch

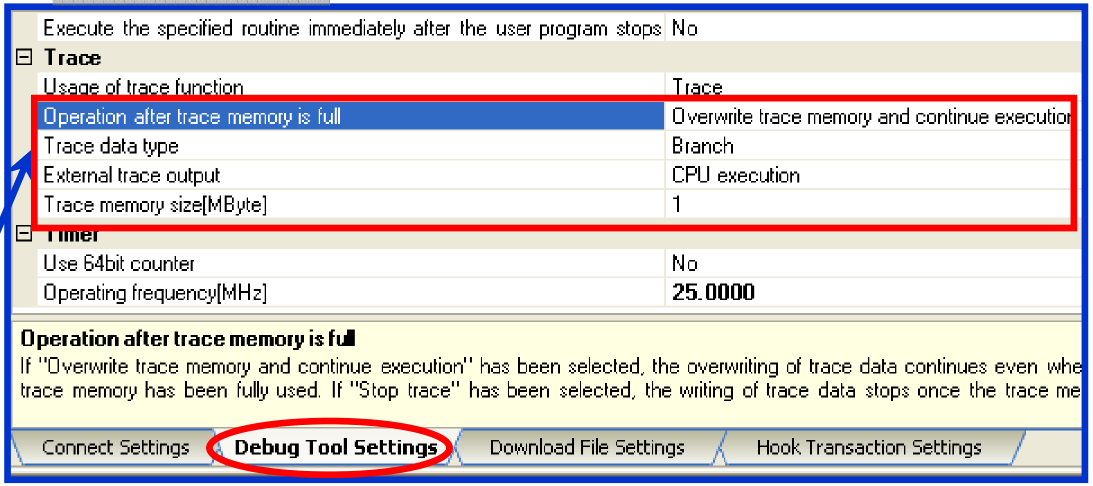
Trace Capacity:

Advanced...

Event used PC: 0 OA: 0 Free PC: 8 OA: 4 Registered events...

Save... Load... Apply Close

CubeSuite+



Execute the specified routine immediately after the user program stops	No
⊟ Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue executio
Trace data type	Branch
External trace output	CPU execution
Trace memory size[MByte]	1
⊟ Timer	
Use 64bit counter	No
Operating frequency[MHz]	**25.0000**

Operation after trace memory is full

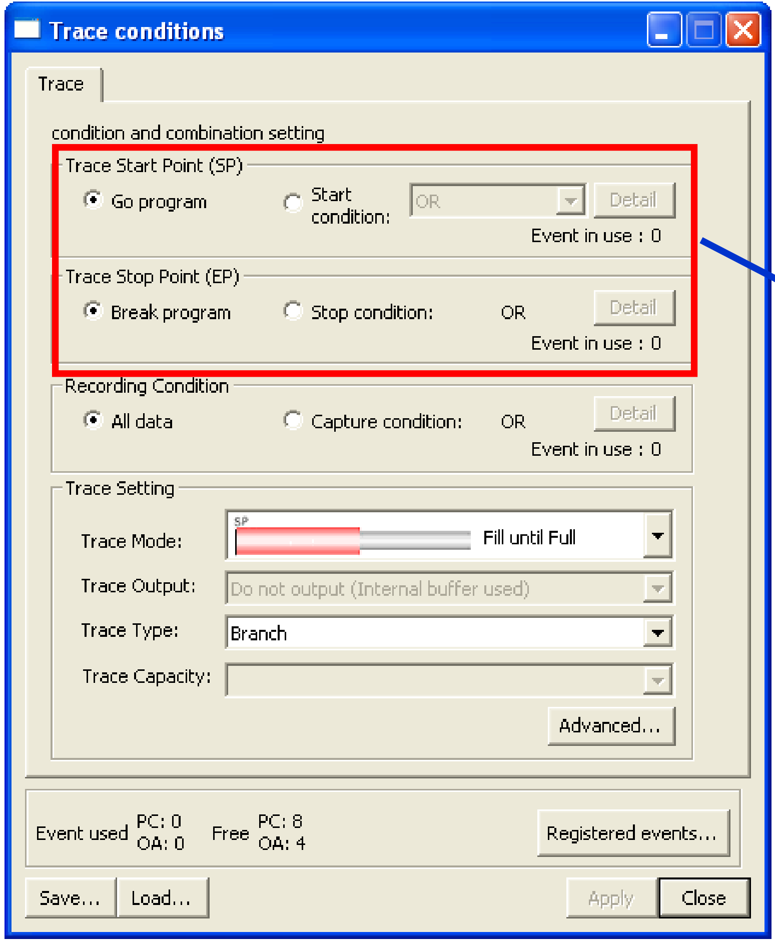
If "Overwrite trace memory and continue execution" has been selected, the overwriting of trace data continues even whe trace memory has been fully used. If "Stop trace" has been selected, the writing of trace data stops once the trace me

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

Note: CubeSuite+ V1.01.00 does not support advanced settings (including timestamps). The next and subsequent versions will support advanced settings.

14. Setting Trace Start/Stop Conditions

14.1 You can use the editor panel to specify an address where trace acquisition will start or stop (see below).



Trace conditions

Trace

condition and combination setting

Trace Start Point (SP)

● Go program ○ Start condition: OR Detail

Event in use : 0

Trace Stop Point (EP)

● Break program ○ Stop condition: OR Detail

Event in use : 0

Recording Condition

● All data ○ Capture condition: OR Detail

Event in use : 0

Trace Setting

Trace Mode: Fill until Full

Trace Output: Do not output (Internal buffer used)

Trace Type: Branch

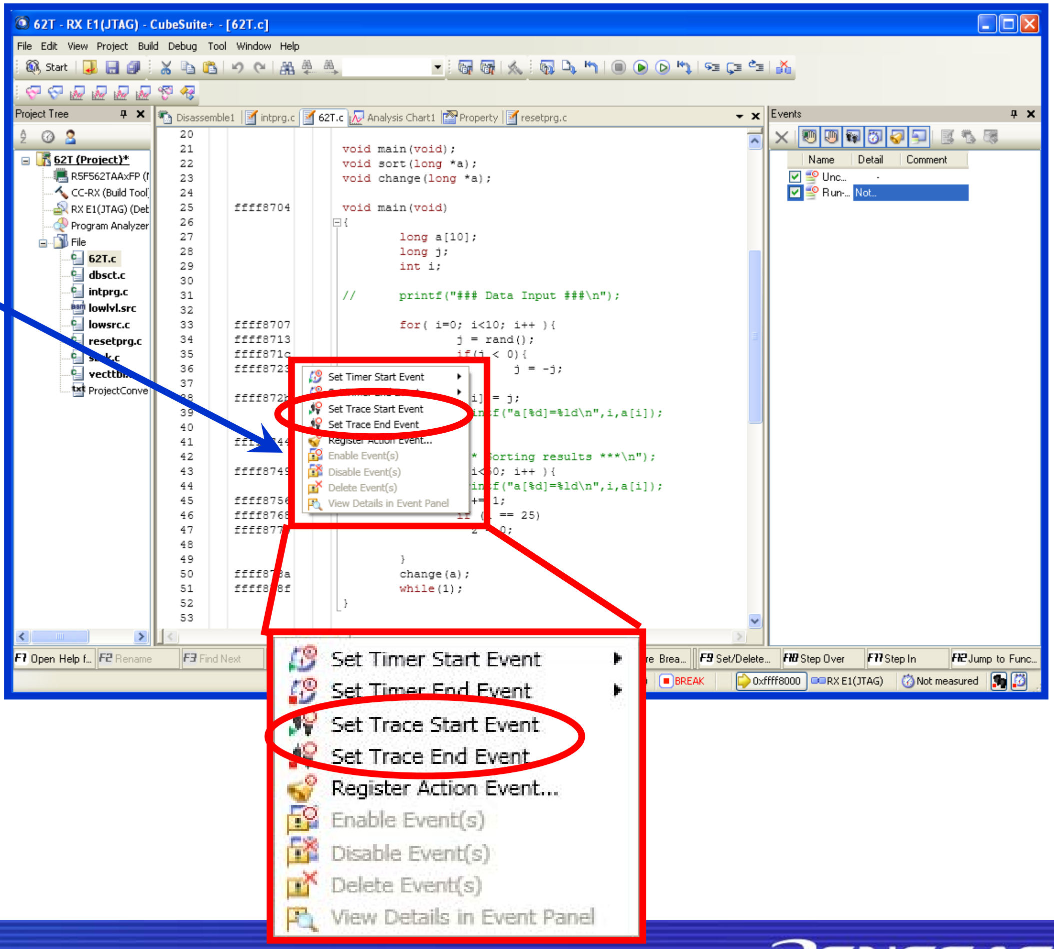
Trace Capacity:

Advanced...

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

Registered events...

Save... Load... Apply Close



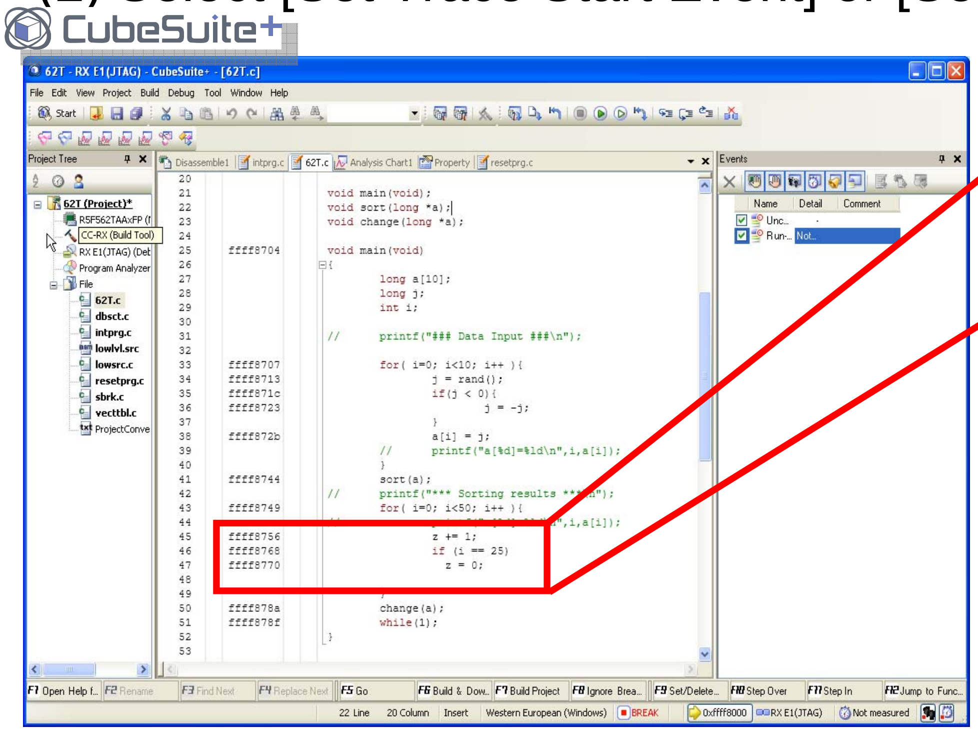
62T - RX E1(JTAG) - CubeSuite+ - [62T.c]

```
                void main(void);
                void sort(long *a);
                void change(long *a);

ffff8704        void main(void)
                {
                        long a[10];
                        long j;
                        int i;

//                      printf("### Data Input ###\n");

ffff8707                for( i=0; i<10; i++ ){
ffff8713                        j = rand();
ffff871c                        if(j < 0){
ffff8723                                j = -j;

                        }
                        change(a);
                        while(1);
                }
```

14. Setting Trace Start/Stop Conditions

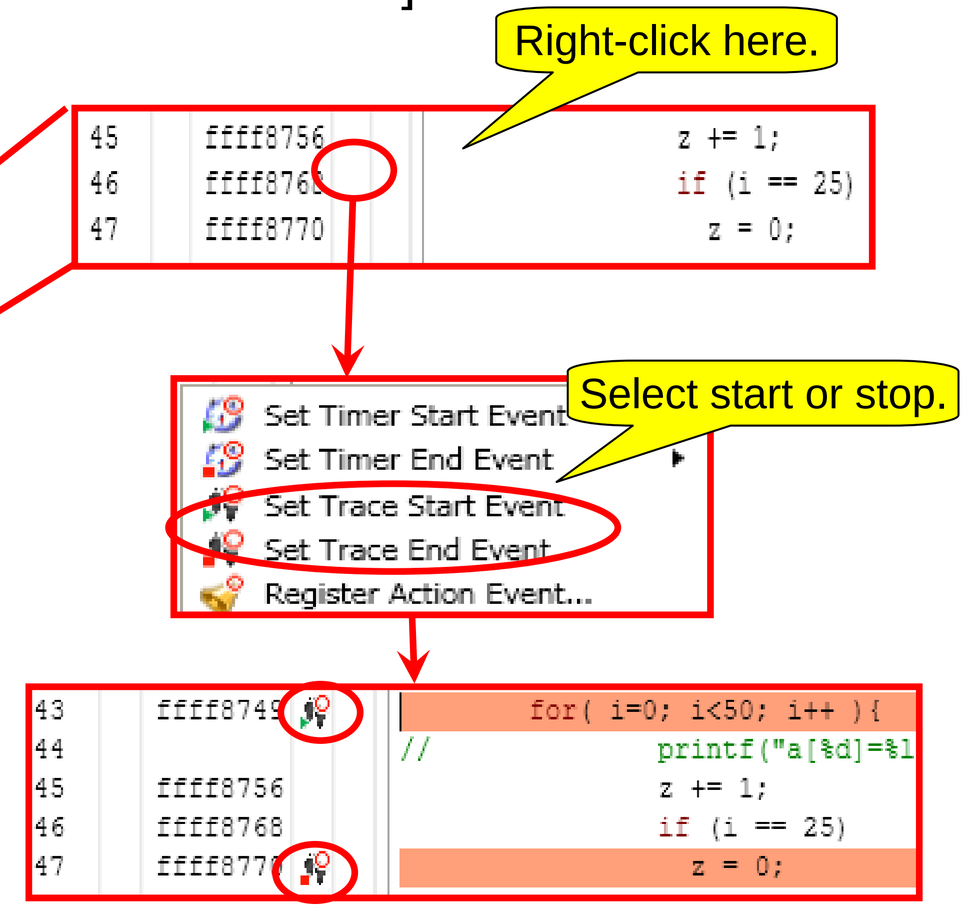
(1) Right-click the event area of the line at which you want to start or stop tracing. This opens a pop-up menu.

(2) Select [Set Trace Start Event] or [Set Trace End Event].



CubeSuite+
62T - RX E1(JTAG) - CubeSuite+ - [62T.c]
File Edit View Project Build Debug Tool Window Help
Project Tree
Disassemble1 | intprg.c | 62T.c | Analysis Chart1 | Property | resetprg.c
Events
Name Detail Comment
F5 Go
F6 Build & Dow...
F7 Build Project
F8 Ignore Brea...
F9 Set/Delete...
F10 Step Over
F11 Step In
F12 Jump to Func...
22 Line 20 Column Insert Western European (Windows) BREAK 0xffff8000 RX E1(JTAG) Not measured

```
                void main(void);
                void sort(long *a);
                void change(long *a);

    ffff8704    void main(void)
                {
                        long a[10];
                        long j;
                        int i;

                //      printf("### Data Input ###\n");

    ffff8707            for( i=0; i<10; i++ ){
    ffff8713                    j = rand();
    ffff871c                    if(j < 0){
    ffff8723                            j = -j;
                                }
    ffff872b                    a[i] = j;
                        //      printf("a[%d]=%ld\n",i,a[i]);
                        }
    ffff8744            sort(a);
                //      printf("*** Sorting results ***\n");
    ffff8749            for( i=0; i<50; i++ ){
                //              printf("a[%d]=%ld\n",i,a[i]);
    ffff8756                    z += 1;
    ffff8768                    if (i == 25)
    ffff8770                      z = 0;

                        }
    ffff878a            change(a);
    ffff878f            while(1);
                }
```



Right-click here.

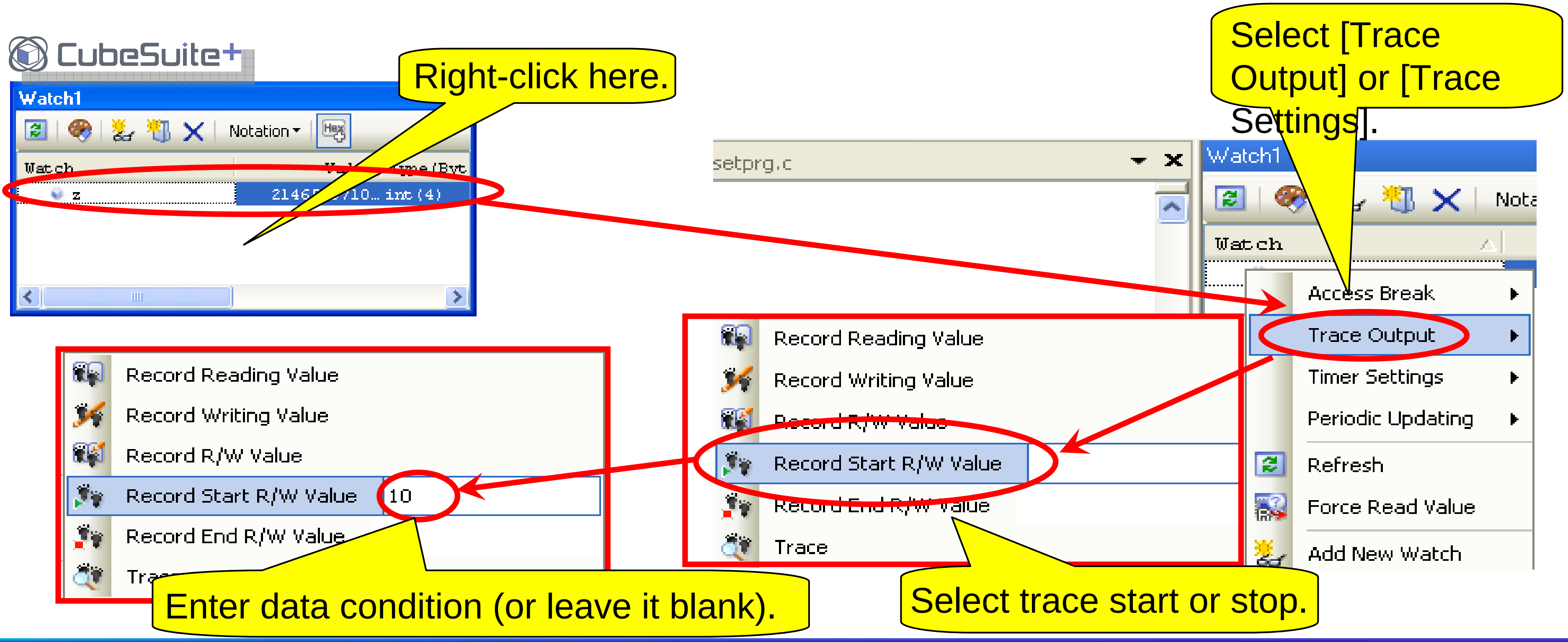
Select start or stop.

```
43    ffff8749    for( i=0; i<50; i++ ){
44                //      printf("a[%d]=%l
45    ffff8756            z += 1;
46    ffff8768            if (i == 25)
47    ffff8770              z = 0;
```

(3) is displayed in the line for which [Set Trace Start Event] has been set, and is displayed in the line for which [Set Trace Stop Event] has been set.

14. Setting Trace Start/Stop Conditions

14.2 You can use the watch or editor panel to start or stop tracing on data access.
(1) On the watch or editor panel, right-click the variable that you want to start or stop tracing when it is accessed. This opens a pop-up menu.
(2) Select [Trace Output] (or [Trace Settings] on the editor panel) and select [Record Start R/W Value] or [Record End R/W Value].
(3) Enter a value to set a data condition.



Right-click here.

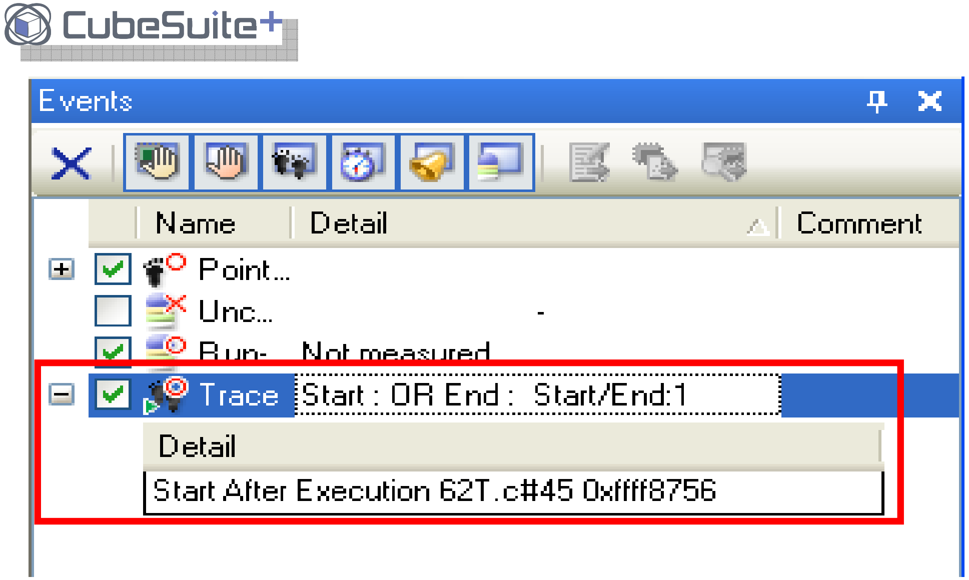
Select [Trace Output] or [Trace Settings].

Select trace start or stop.

Enter data condition (or leave it blank).

14. Setting Trace Start/Stop Conditions

All conditions that have been set can be checked on the event panel.

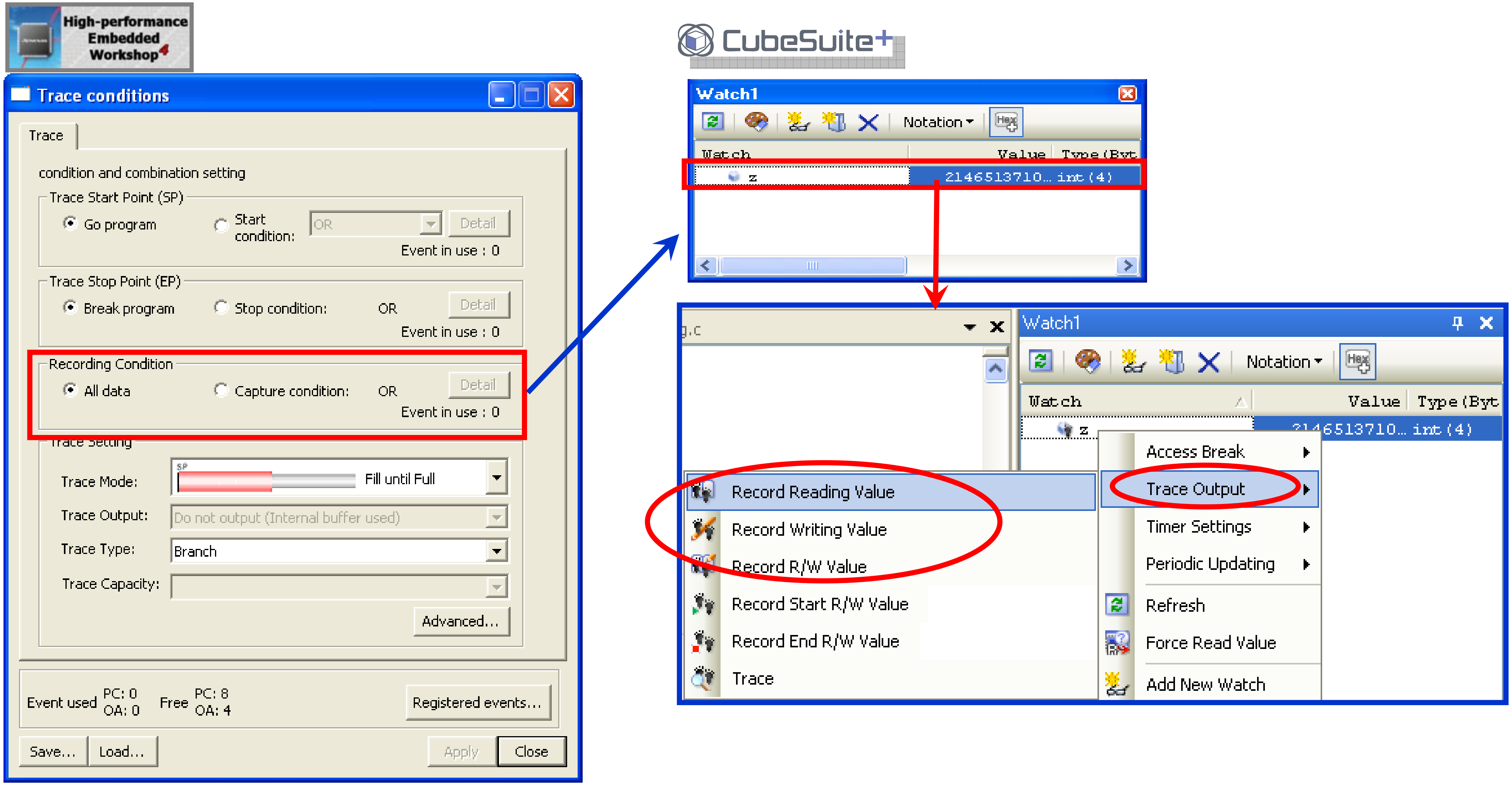


Name	Detail	Comment
☑ Point...		
□ Unc...	-	
☑ Run-	Not measured	
☑ Trace	Start : OR End : Start/End:1	

Detail
Start After Execution 62T.c#45 0xffff8756

15. Setting Trace Extraction Conditions

You can set trace extraction conditions on the watch or editor panel (see below).



High-performance Embedded Workshop

Trace conditions

Trace

condition and combination setting

Trace Start Point (SP)
● Go program ○ Start condition: OR Detail
Event in use : 0

Trace Stop Point (EP)
● Break program ○ Stop condition: OR Detail
Event in use : 0

Recording Condition
● All data ○ Capture condition: OR Detail
Event in use : 0

Trace Setting
Trace Mode: Fill until Full
Trace Output: Do not output (Internal buffer used)
Trace Type: Branch
Trace Capacity:
Advanced...

Event used PC: 0 OA: 0 Free PC: 8 OA: 4 Registered events...

Save... Load... Apply Close

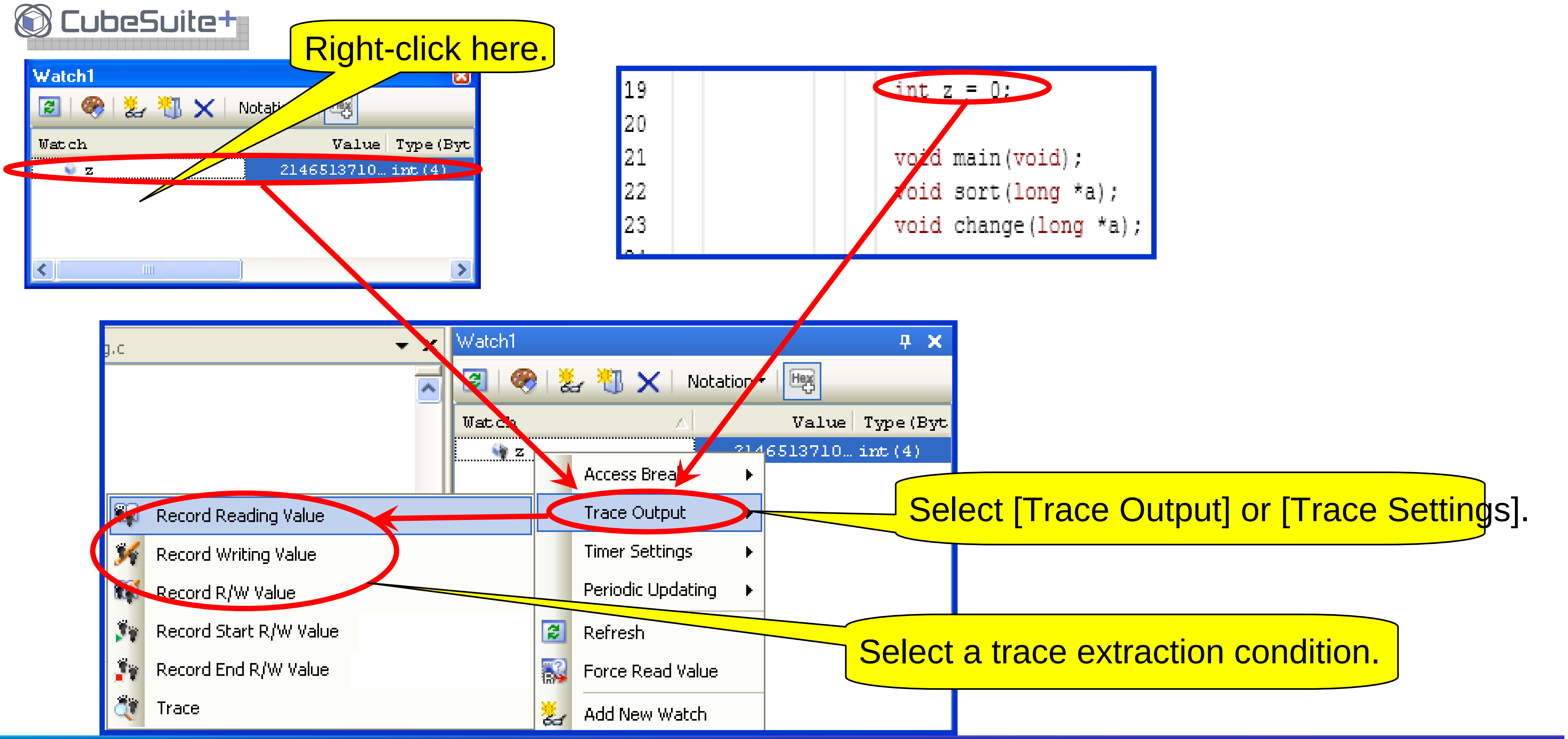
CubeSuite+

Watch1

Watch	Value	Type (Byt
z	2146513710...	int (4)

15. Setting Trace Extraction Conditions

(1) On the watch or editor panel, right-click the variable that you want to use as a trace extraction condition. This opens a pop-up menu.
(2) Select [Trace Output] (or [Trace Settings] on the editor panel) and select a trace extraction condition.



Right-click here.

Watch	Value	Type (Byt
z	2146513710…	int (4)

```
int z = 0;

void main(void);
void sort(long *a);
void change(long *a);
```

Select [Trace Output] or [Trace Settings].

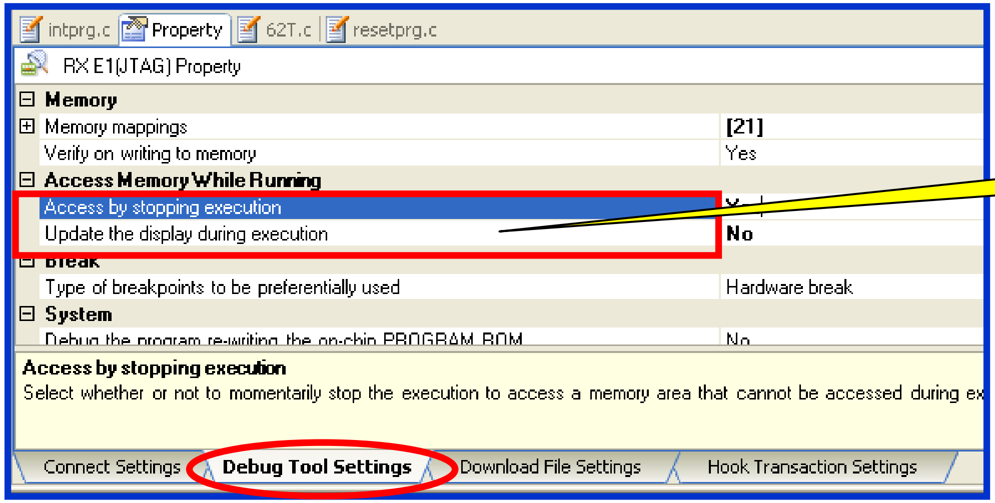
Select a trace extraction condition.

16. Viewing/Changing Memory Data and Variables While the Program Is Running

To view or change memory data and variables while the program is running in CubeSuite+, make settings on the [Properties] panel by using the following procedure:

(1) Open the [Debug Tool Settings] sheet on the [Properties] panel of the debug tool.
(2) Set [Access by stopping execution] in the [Access Memory While Running] field to [Yes]. Memory data and variables can be viewed while the program is running.

CubeSuite+



RX E1(JTAG) Property

⊟ **Memory**	
⊞ Memory mappings	[21]
Verify on writing to memory	Yes
⊟ **Access Memory While Running**	
Access by stopping execution	Y
Update the display during execution	**No**
⊟ **Break**	
Type of breakpoints to be preferentially used	Hardware break
⊟ **System**	
Debug the program re-writing the on-chip PROGRAM ROM	No

Access by stopping execution
Select whether or not to momentarily stop the execution to access a memory area that cannot be accessed during ex

Change this to [Yes].

If [No] is selected, “**” is displayed on the memory panel while the program is running.

17. Automatically Updating Memory Data and Variables While the Program Is Running

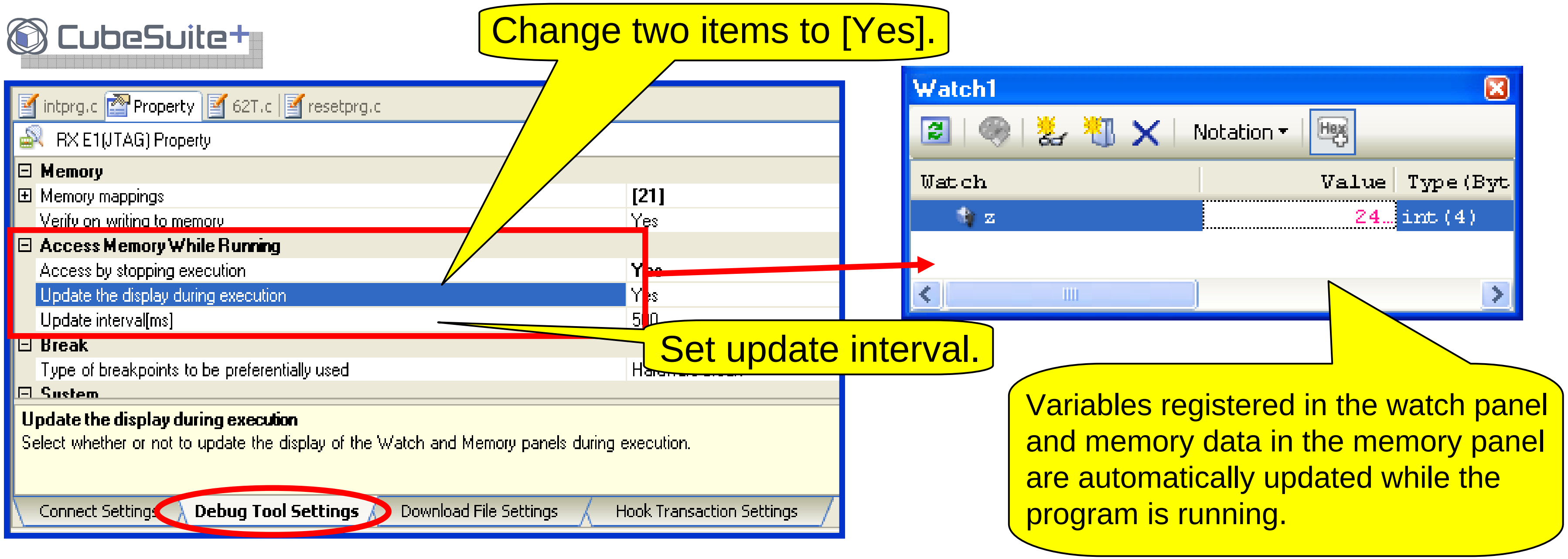
To automatically update memory data and variables via CubeSuite+, make settings on the [Properties] panel by using the following procedure:

(1) Open the [Debug Tool Settings] sheet on the [Properties] panel of the debug tool.

(2) Set [Access by stopping execution] and [Update the display during execution] in the [Access Memory While Running] field to [Yes].

Information displayed on the memory and watch panels is automatically updated while the program is running.

To change the update interval, modify the [Update interval] value.



Change two items to [Yes].

RX E1(JTAG) Property

Memory
Memory mappings [21]
Verify on writing to memory Yes
Access Memory While Running
Access by stopping execution Yes
Update the display during execution Yes
Update interval[ms] 500
Break
Type of breakpoints to be preferentially used Hardware break
System

Update the display during execution
Select whether or not to update the display of the Watch and Memory panels during execution.

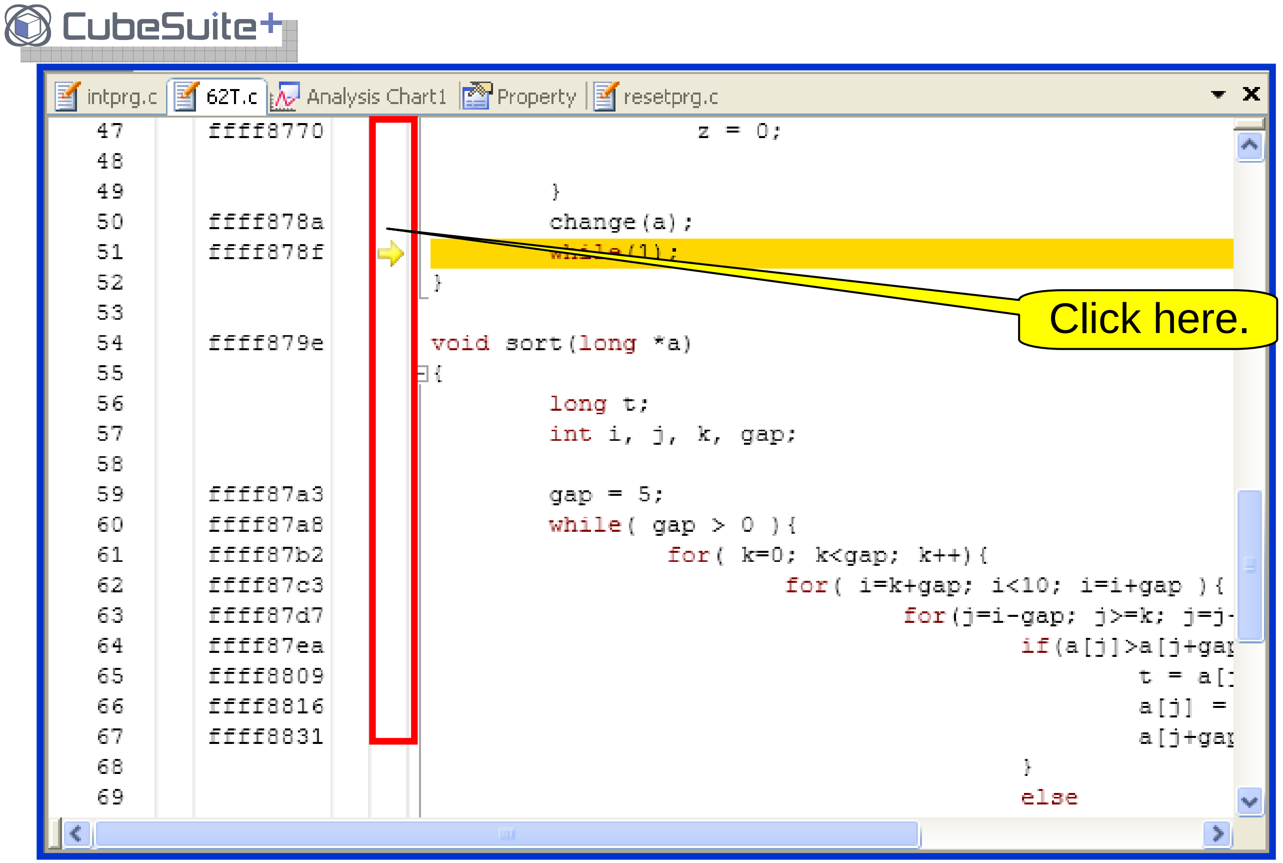
Set update interval.

Watch	Value	Type (Byt
z	24...	int (4)

Variables registered in the watch panel and memory data in the memory panel are automatically updated while the program is running.

18. Setting Breakpoints

(1) You can set breakpoints in the main area (enclosed by a red line in the figure below) on the editor panel of CubeSuite+.
Single-clicking a line with an address sets a breakpoint.
Single-clicking a line for which a breakpoint has been set deletes the breakpoint.

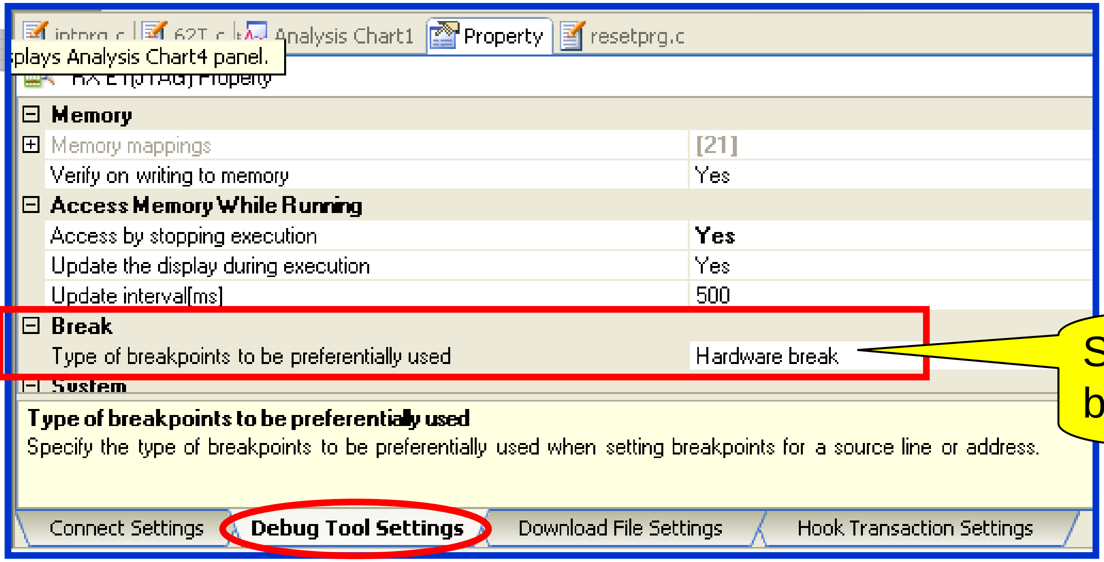


```
ffff8770             z = 0;

             }
ffff878a     change(a);
ffff878f     while(1);
         }

ffff879e void sort(long *a)
         {
             long t;
             int i, j, k, gap;

ffff87a3     gap = 5;
ffff87a8     while( gap > 0 ){
ffff87b2         for( k=0; k<gap; k++){
ffff87c3             for( i=k+gap; i<10; i=i+gap ){
ffff87d7                 for(j=i-gap; j>=k; j=j-
ffff87ea                     if(a[j]>a[j+gap
ffff8809                         t = a[j
ffff8816                         a[j] =
ffff8831                         a[j+gap
                             }
                             else
```

Click here.

18. Setting Breakpoints

(2) Select a breakpoint type (software break or hardware break) for [Type of breakpoints to be preferentially used] in the [Debug Tool Settings] sheet on the [Properties] panel. (Hardware break is selected in the example below.)



CubeSuite+

⊟ **Memory**	
⊞ Memory mappings	[21]
Verify on writing to memory	Yes
⊟ **Access Memory While Running**	
Access by stopping execution	**Yes**
Update the display during execution	Yes
Update interval[ms]	500
⊟ **Break**	
Type of breakpoints to be preferentially used	Hardware break
⊟ **System**	

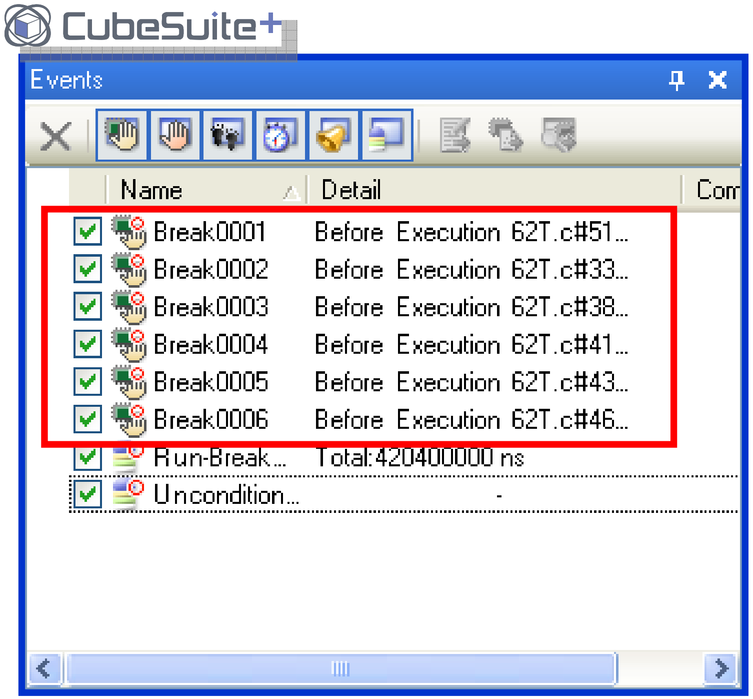
Type of breakpoints to be preferentially used
Specify the type of breakpoints to be preferentially used when setting breakpoints for a source line or address.

Set a breakpoint type to be preferentially used.

(3) If the number of breakpoints of the selected type exceeds the limit, the other type of breakpoints are used.
Event marks indicate the types of breakpoints.
: Software break : Hardware break

18. Setting Breakpoints

(4) You can check the breakpoint setting on the [Event] panel.
Select [View] -> [Event] from the CubeSuite+ menu to open the [Event] panel.
Unnecessary breakpoints can be deleted or disabled on the [Event] panel.



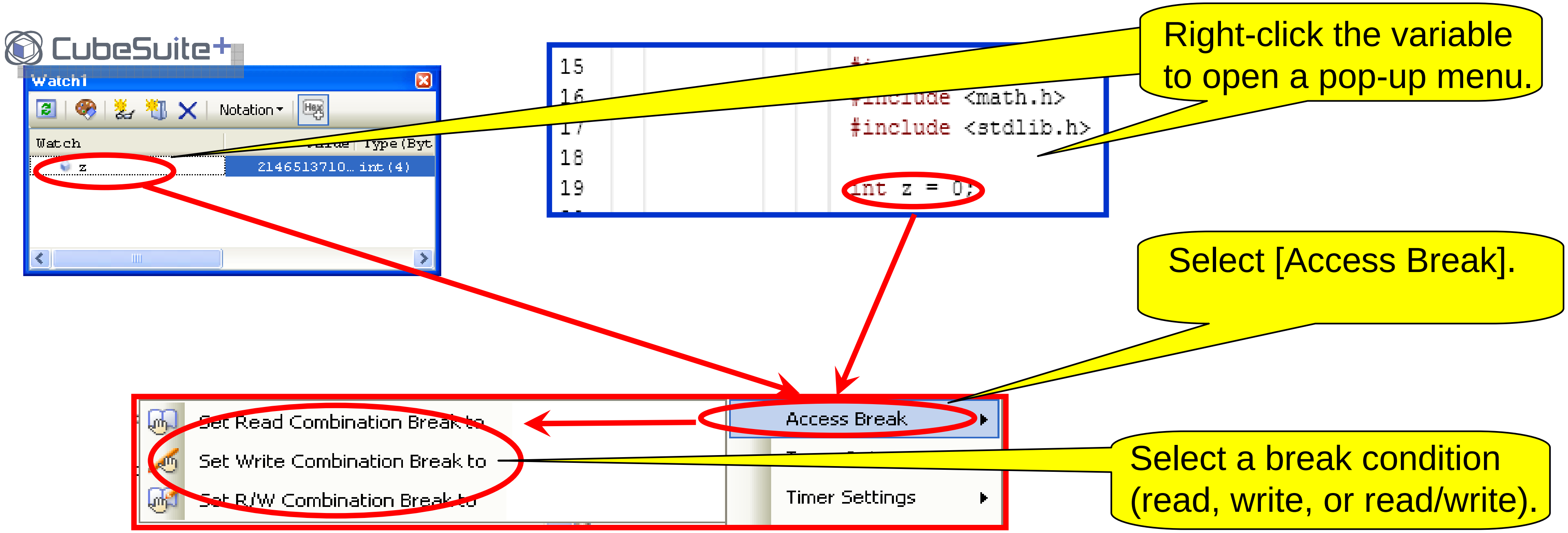
	Name	Detail	Com
☑	Break0001	Before Execution 62T.c#51...	
☑	Break0002	Before Execution 62T.c#33...	
☑	Break0003	Before Execution 62T.c#38...	
☑	Break0004	Before Execution 62T.c#41...	
☑	Break0005	Before Execution 62T.c#43...	
☑	Break0006	Before Execution 62T.c#46...	
☑	Run-Break...	Total:420400000 ns	
☑	Uncondition...	-	

19. Causing a Break on Access to a Variable

You can use the watch or editor panel to make a setting to cause a break on access to a specific variable.

(1) On the watch or editor panel, right-click the variable that you want to cause a break when it is accessed. This opens a pop-up menu.

(2) Select [Access Break] (or [Break Settings] on the editor panel) and select [Set Read Combination Break to], [Set Write Combination Break to], or [Set R/W Combination Break to].



```
#include <math.h>
#include <stdlib.h>

int z = 0;
```

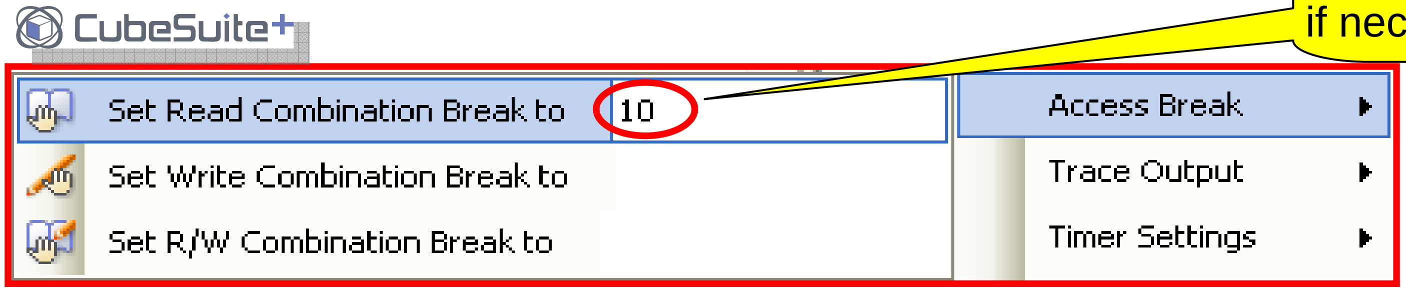
Right-click the variable to open a pop-up menu.

Select [Access Break].

Select a break condition (read, write, or read/write).

19. Causing a Break on Access to a Variable

(3) Enter a value to set a data condition (or leave the box blank if no data condition is needed).


CubeSuite+
Set Read Combination Break to
10
Set Write Combination Break to
Set R/W Combination Break to
Access Break
Trace Output
Timer Settings

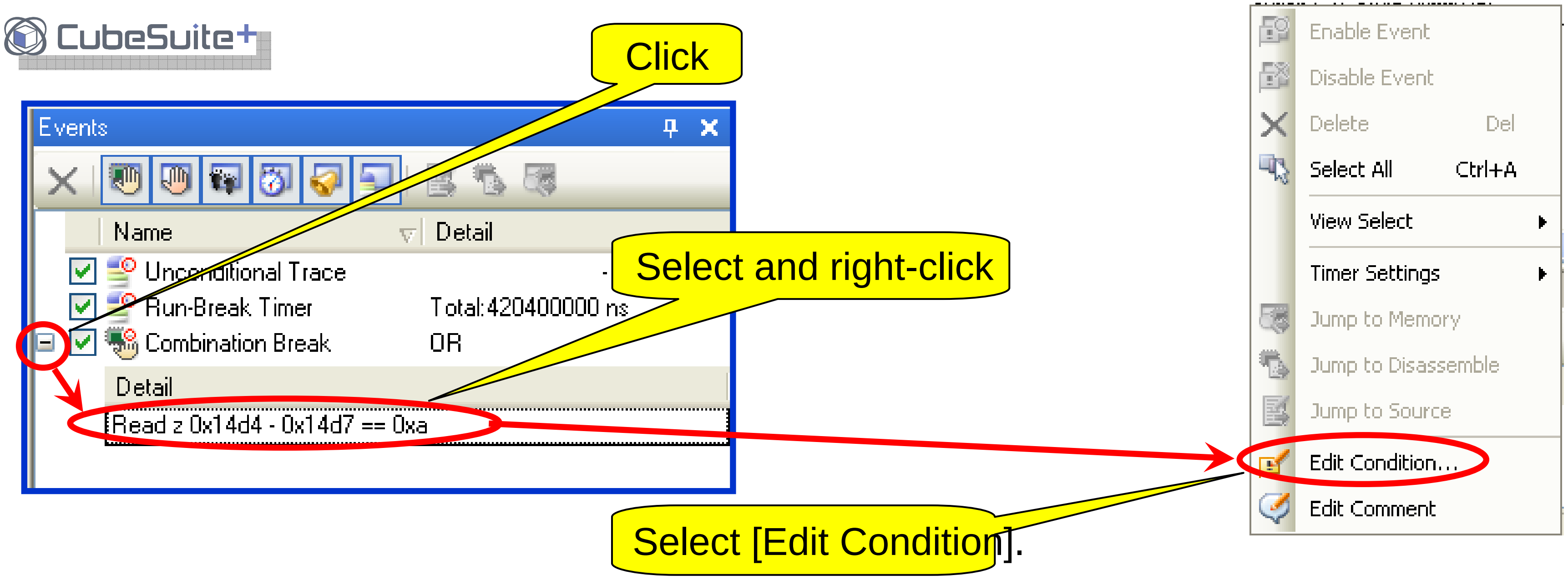
Enter a data condition if necessary.

Note: Enter a decimal number here. When entering a hexadecimal number, add “0x” to the head (e.g. 0xAA).

19. Causing a Break on Access to a Variable

You can also specify masking of data conditions in the [Detailed Settings of Access Events] dialog box.

(4) Select [View] -> [Event] from the menu to open the event panel. Click the “+” mark to the left of [Combination Break] for expansion.
(5) Select and right-click the event you want to modify in the [Detail] field to open a pop-up menu. Select [Edit Condition] from the pop-up menu.



Name	Detail
☑ Unconditional Trace	-
☑ Run-Break Timer	Total:420400000 ns
☑ Combination Break	OR

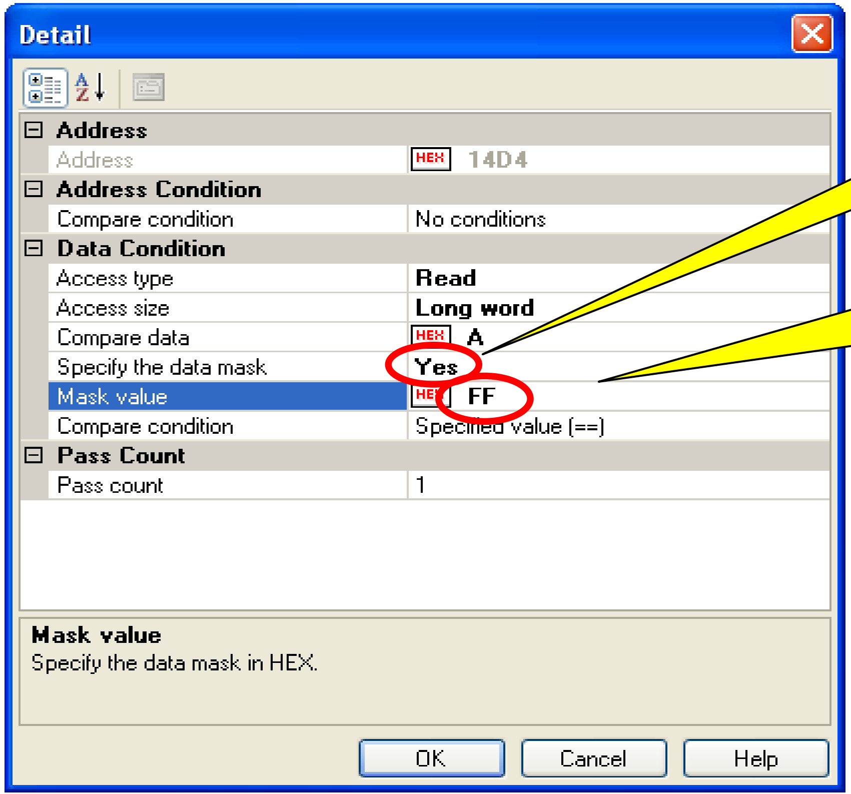
Detail

Read z 0x14d4 - 0x14d7 == 0xa

19. Causing a Break on Access to a Variable

(6) The [Detailed Settings of Access Events] dialog box opens. Set [Specify the data mask] in the [Data Condition] field to [Yes].
(7) Setting of a mask value is enabled. Enter a mask value.
The bits of the data value for which a mask value “0” has been specified is treated as “Don’t Care”.

CubeSuite+



Detail

Address	
Address	HEX 14D4
Address Condition	
Compare condition	No conditions
Data Condition	
Access type	Read
Access size	Long word
Compare data	HEX A
Specify the data mask	Yes
Mask value	HEX FF
Compare condition	Specified value (==)
Pass Count	
Pass count	1

Mask value
Specify the data mask in HEX.

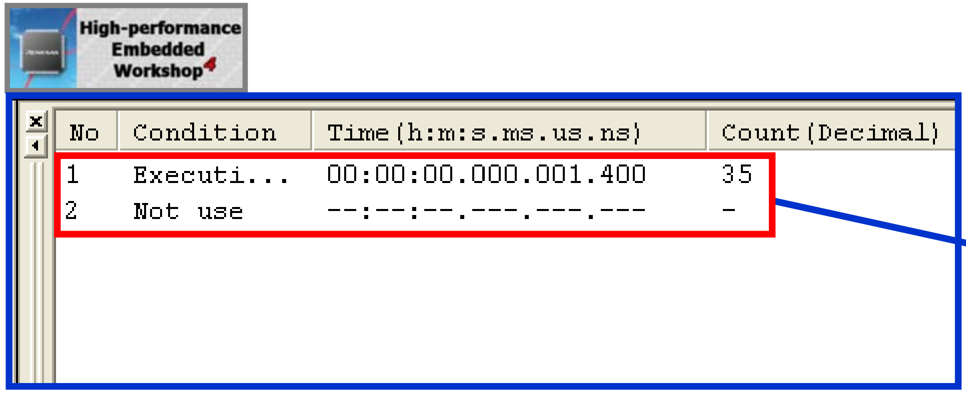
Set this item to [Yes].

Enter a mask value in hexadecimal.
This example shows masking (Don’t care) of bytes other than the lowest 1 byte.

This dialog box allows settings for masking address conditions, specifying a range, access type, access size, comparison data, and pass count for data conditions in addition to masking data conditions.

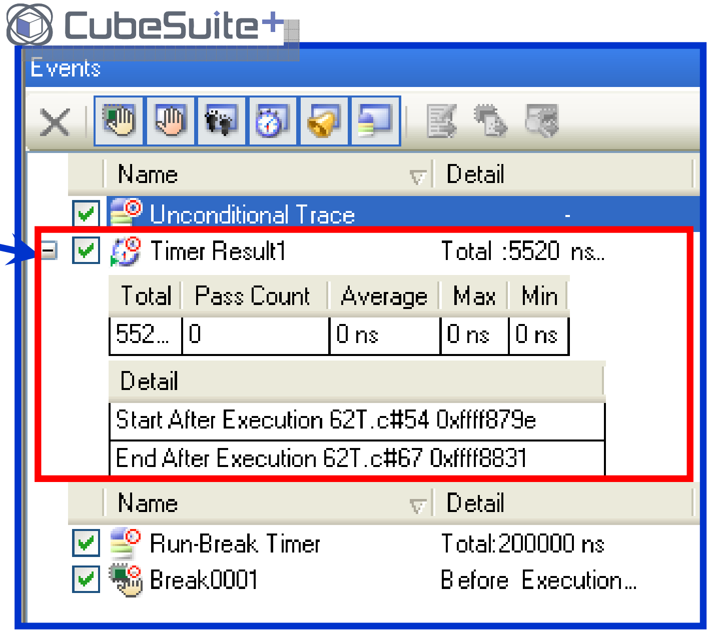
20. Using the Performance-Measurement (Timer) Function

While the HEW can measure program execution time with the performance-measurement function, the CubeSuite+ allows measurement of program execution time by the timer.
Measurement results can be viewed on the [Event] panel (opened by selecting [View] -> [Event] from the menu).



No	Condition	Time (h:m:s.ms.us.ns)	Count (Decimal)
1	Executi...	00:00:00.000.001.400	35
2	Not use	--:--:--.---.---.---	-

Performance Window



Name	Detail
☑ Unconditional Trace	-
☑ Timer Result1	Total :5520 ns...

Total	Pass Count	Average	Max	Min
552...	0	0 ns	0 ns	0 ns

Detail
Start After Execution 62T.c#54 0xffff879e
End After Execution 62T.c#67 0xffff8831

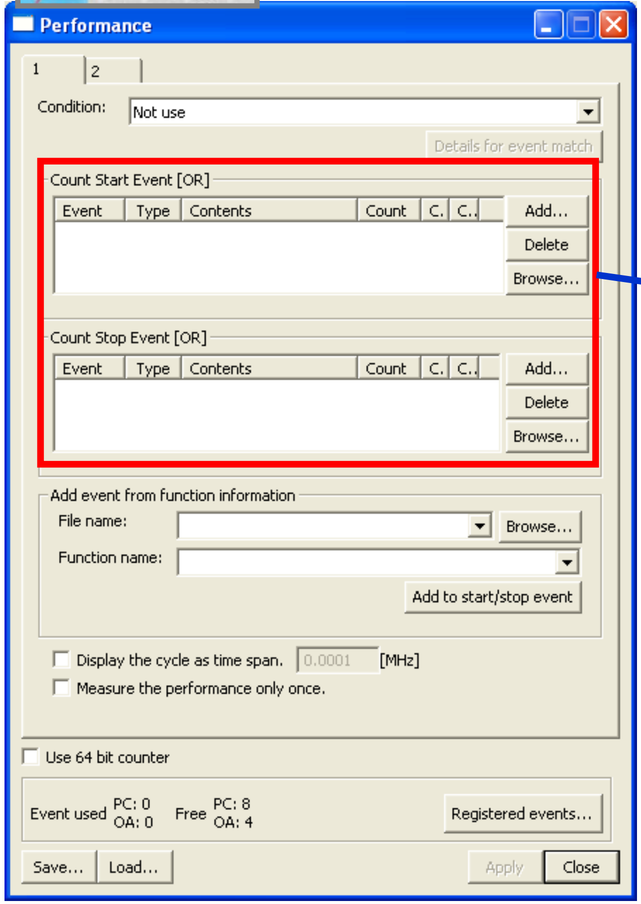
Name	Detail
☑ Run-Break Timer	Total:200000 ns
☑ Break0001	Before Execution...

[Event] Panel

20. Using the Performance-Measurement (Timer) Function

20.1 You can set measurement start/stop conditions on the editor panel.
The procedure is described on the next page.

High-performance Embedded Workshop

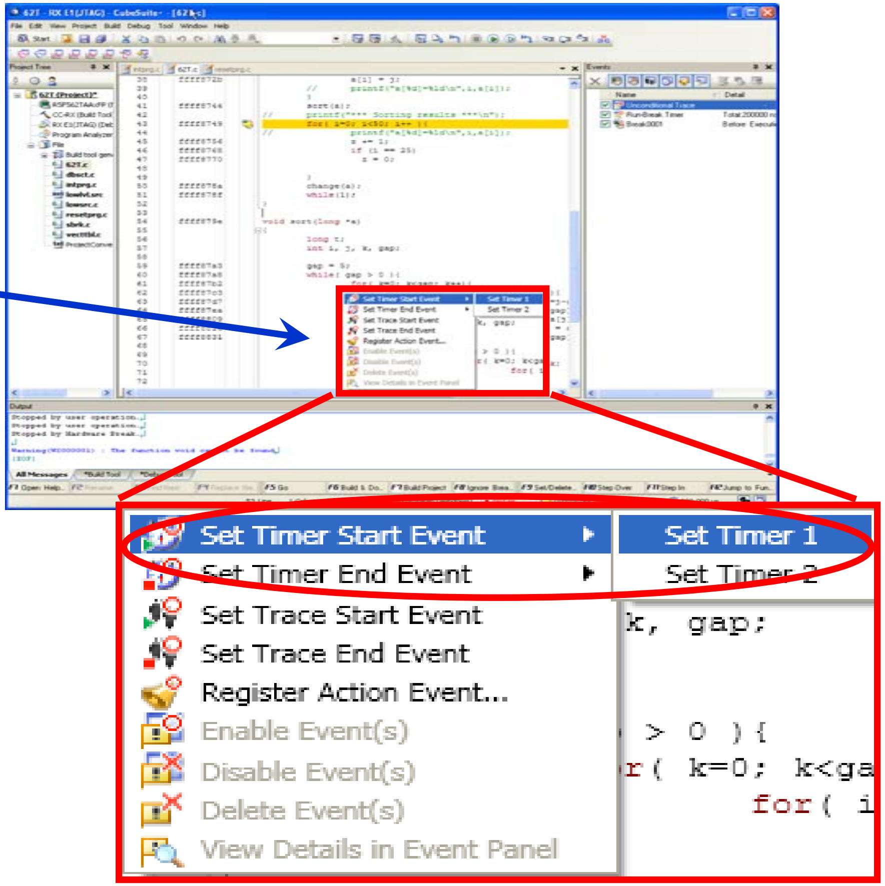


Condition: Not use

Display the cycle as time span. 0.0001 [MHz]

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

CubeSuite+



Note: Start/stop conditions cannot be registered through function names in CubeSuite+.

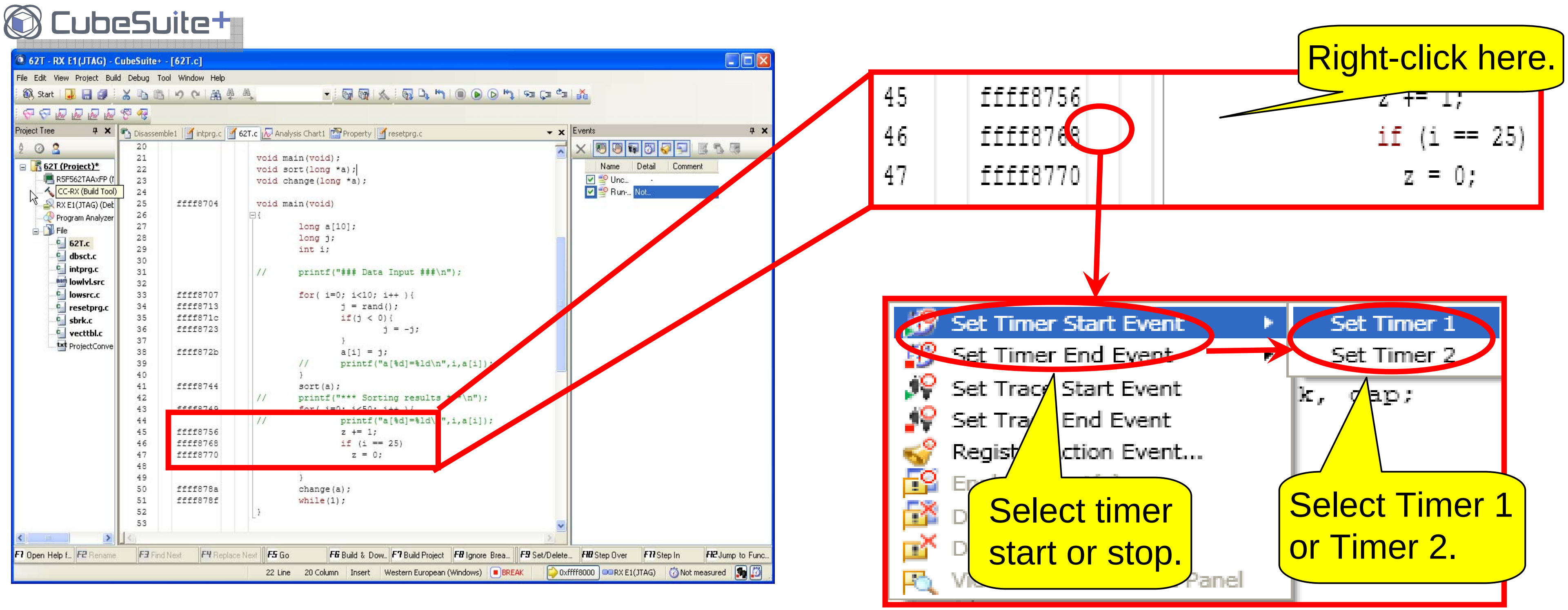
20. Using the Performance-Measurement (Timer) Function

Set measurement start/stop conditions by using the following procedure.

(1) On the [Editor] panel, right-click the event area of the line at which you want to start or stop measurement. This opens a pop-up menu.

(2) Select [Set Timer Start Event] or [Set Timer End Event].

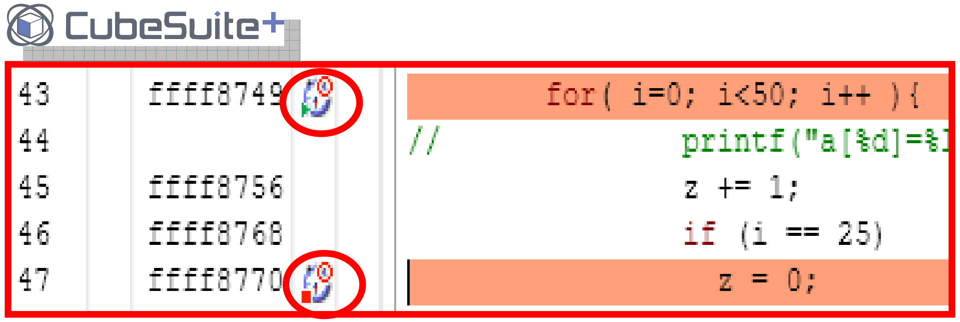
(3) If two channels are provided, select a timer (Timer 1 or Timer 2) to be used for measurement.



```
                  void main(void);
                  void sort(long *a);
                  void change(long *a);

ffff8704          void main(void)
                  {
                          long a[10];
                          long j;
                          int i;

                  //      printf("### Data Input ###\n");

ffff8707                  for( i=0; i<10; i++ ){
ffff8713                          j = rand();
ffff871c                          if(j < 0){
ffff8723                                  j = -j;
                                  }
ffff872b                          a[i] = j;
                  //              printf("a[%d]=%ld\n",i,a[i]);
                          }
ffff8744                  sort(a);
                  //      printf("*** Sorting results ***\n");
ffff8749                  for( i=0; i<50; i++ ){
                  //              printf("a[%d]=%ld\n",i,a[i]);
ffff8756                          z += 1;
ffff8768                          if (i == 25)
ffff8770                            z = 0;

                          }
ffff878a                  change(a);
ffff878f                  while(1);
                  }
```

20. Using the Performance-Measurement (Timer) Function

(4) Upon completion of the setting, ⏱ appears in the line for which [Set Timer Start Event] has been set, and ⏱ appears in the line for which [Set Timer End Event] has been set.

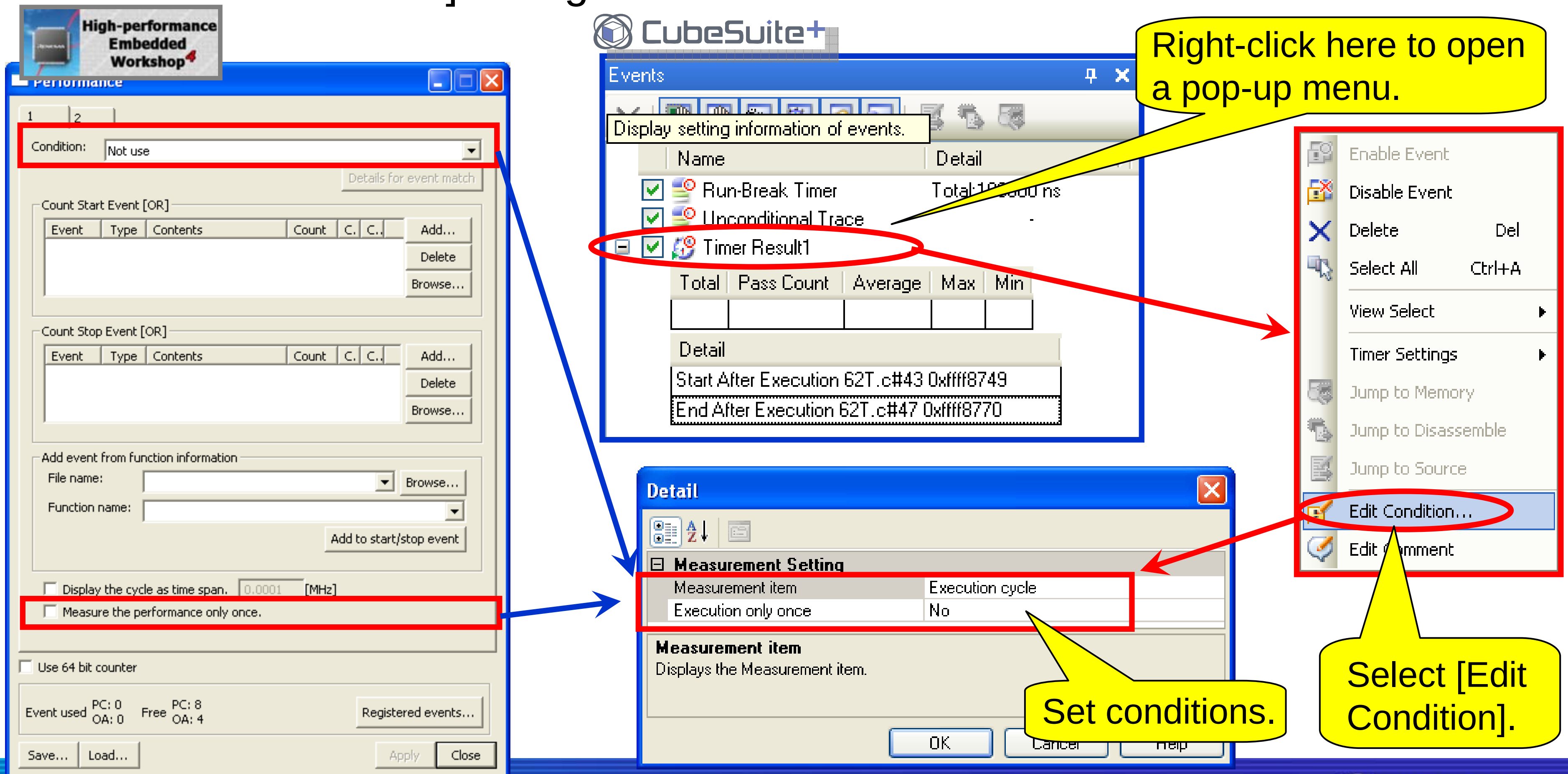


```
ffff8749        for( i=0; i<50; i++ ){
        //              printf("a[%d]=%
ffff8756                z += 1;
ffff8768                if (i == 25)
ffff8770                  z = 0;
```

20. Using the Performance-Measurement (Timer) Function

20.2 You can set [Measurement item] and [Execution only once] on the [Detailed Settings of Timer Measurement] dialog box.

(1) Right-click [Timer Result1] on the [Event] panel to open a pop-up menu. Selecting [Edit Condition] from the pop-up menu opens the [Detailed Settings of Timer Measurement] dialog box.



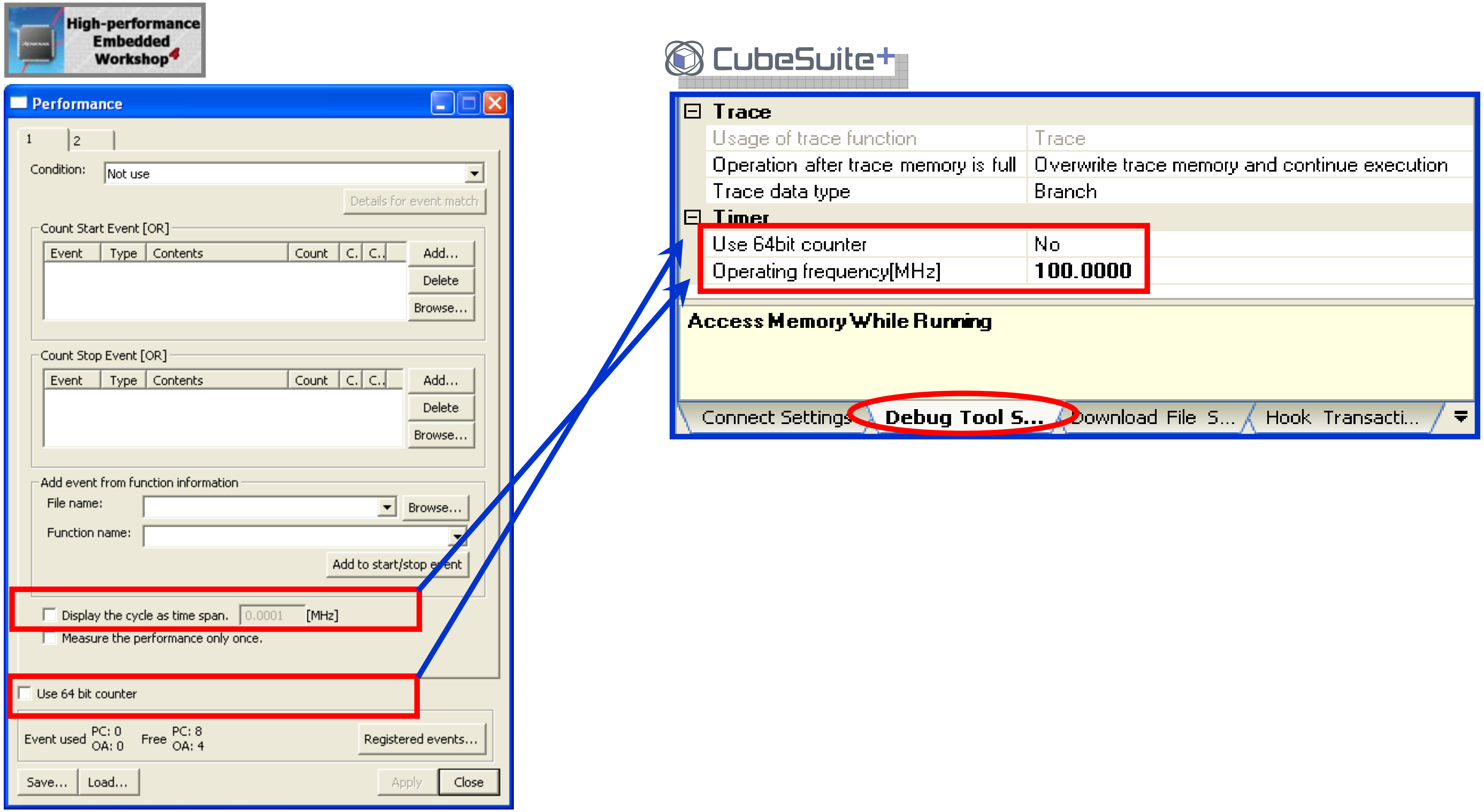
Right-click here to open a pop-up menu.

Select [Edit Condition].

Set conditions.

20. Using the Performance-Measurement (Timer) Function

20.3 You can set [Display the cycle as time span] and [Use 64 bit counter] in the［Debug Tool Settings］sheet on the [Properties] panel.



Condition: Not use

☐ Display the cycle as time span. 0.0001 [MHz]

☐ Measure the performance only once.

☐ Use 64 bit counter

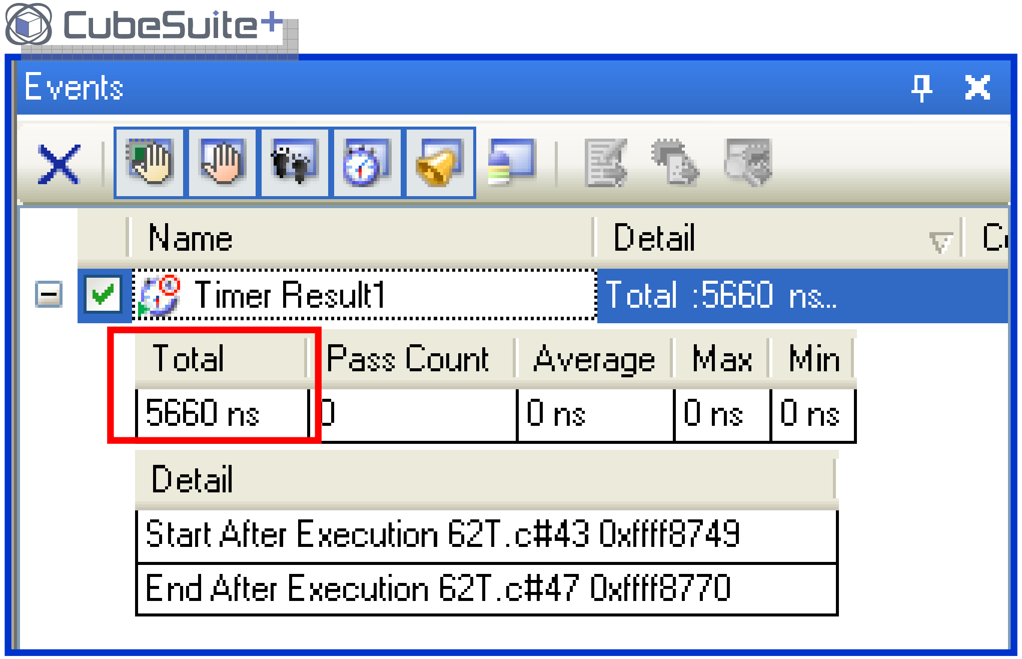
Event used PC: 0 OA: 0 Free PC: 8 OA: 4

⊟ Trace	
Usage of trace function	Trace
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	Branch
⊟ Timer	
Use 64bit counter	No
Operating frequency[MHz]	**100.0000**

Access Memory While Running

20. Using the Performance-Measurement (Timer) Function

20.4 When all settings have been made, execute the program.
The time measurement result is displayed at a break.



Name	Detail
Timer Result1	Total :5660 ns..

Total	Pass Count	Average	Max	Min
5660 ns	0	0 ns	0 ns	0 ns

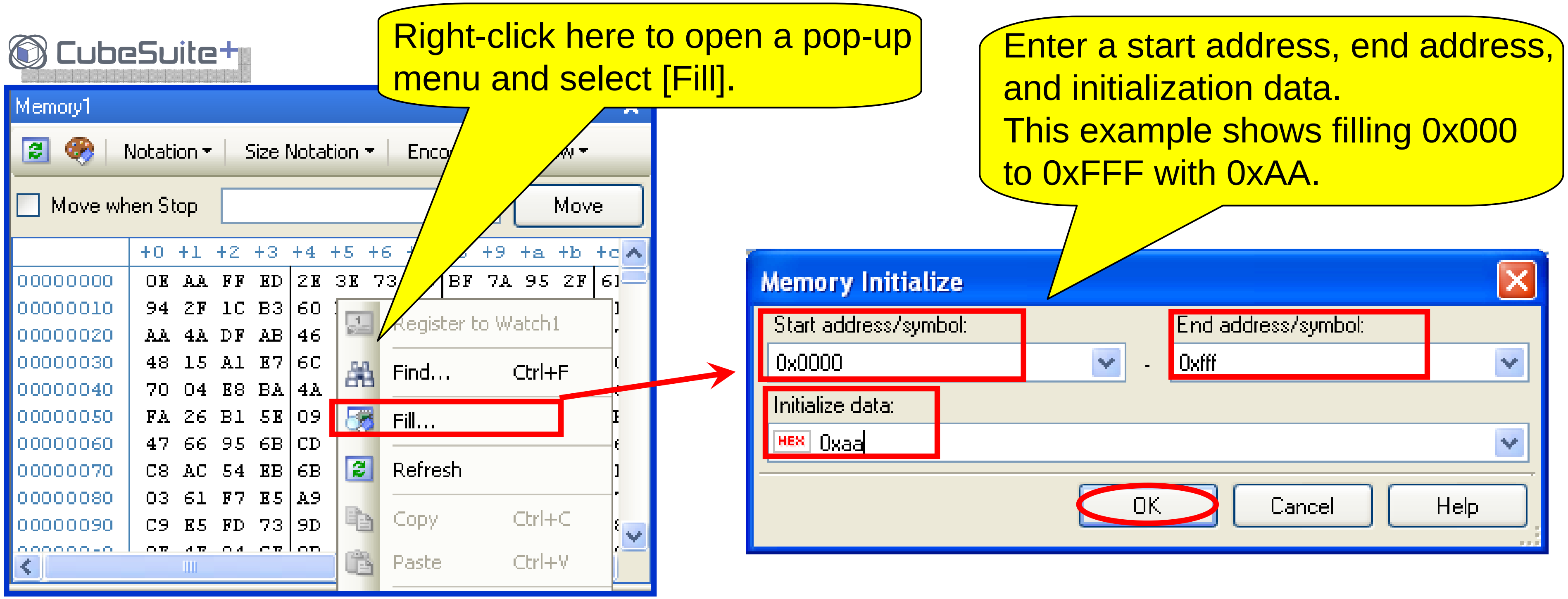
Detail
Start After Execution 62T.c#43 0xffff8749
End After Execution 62T.c#47 0xffff8770

21. Filling Memory

Memory can be filled (batch change) via the [Memory Initialize] dialog box.

(1) Right-click on the [Memory] panel to open a pop-up menu, and select [Fill] from the pop-up menu.

(2) The [Memory Initialize] dialog box opens. Enter addresses (start address and end address) you want to initialize and initialization data, and then click the [OK] button.



```
          +0 +1 +2 +3 +4 +5 +6      +9 +a +b +c
00000000  0E AA FF ED 2E 3E 73      BF 7A 95 2F 61
00000010  94 2F 1C B3 60
00000020  AA 4A DF AB 46
00000030  48 15 A1 E7 6C
00000040  70 04 E8 BA 4A
00000050  FA 26 B1 5E 09
00000060  47 66 95 6B CD
00000070  C8 AC 54 EB 6B
00000080  03 61 F7 E5 A9
00000090  C9 E5 FD 73 9D
```

Start address/symbol: 0x0000

End address/symbol: 0xfff

Initialize data: HEX 0xaa

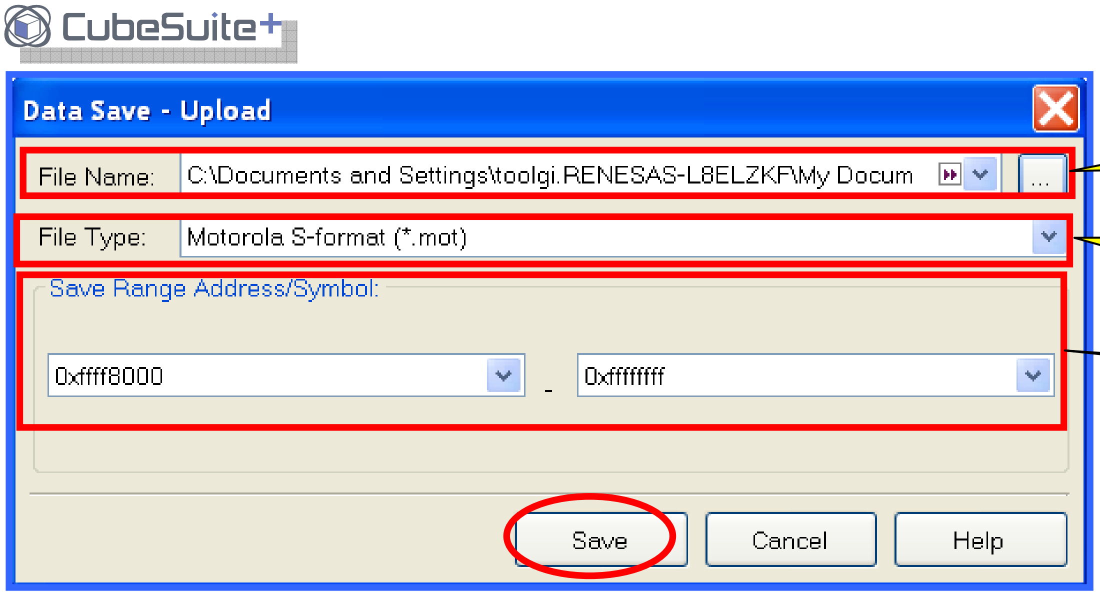
Right-click here to open a pop-up menu and select [Fill].

Enter a start address, end address, and initialization data.
This example shows filling 0x000 to 0xFFF with 0xAA.

Note: Enter decimal numbers here. When entering hexadecimal numbers, add “0x” to the head of each number.

22. Saving Memory Data

[Data Save] dialog box is used to save memory data via CubeSuite+.
Select [Debug] -> [Upload...] from the menu.
The [Data Save] dialog box opens. Specify the file name, type, and range of memory data you want to save, and then click the [Save] button.



File Name: C:\Documents and Settings\toolgi.RENESAS-L8ELZKF\My Docum

File Type: Motorola S-format (*.mot)

Save Range Address/Symbol:

0xffff8000 - 0xffffffff

Enter the name of a file to be saved.

Specify a file type (Intel Hex, Motorola S, or binary).

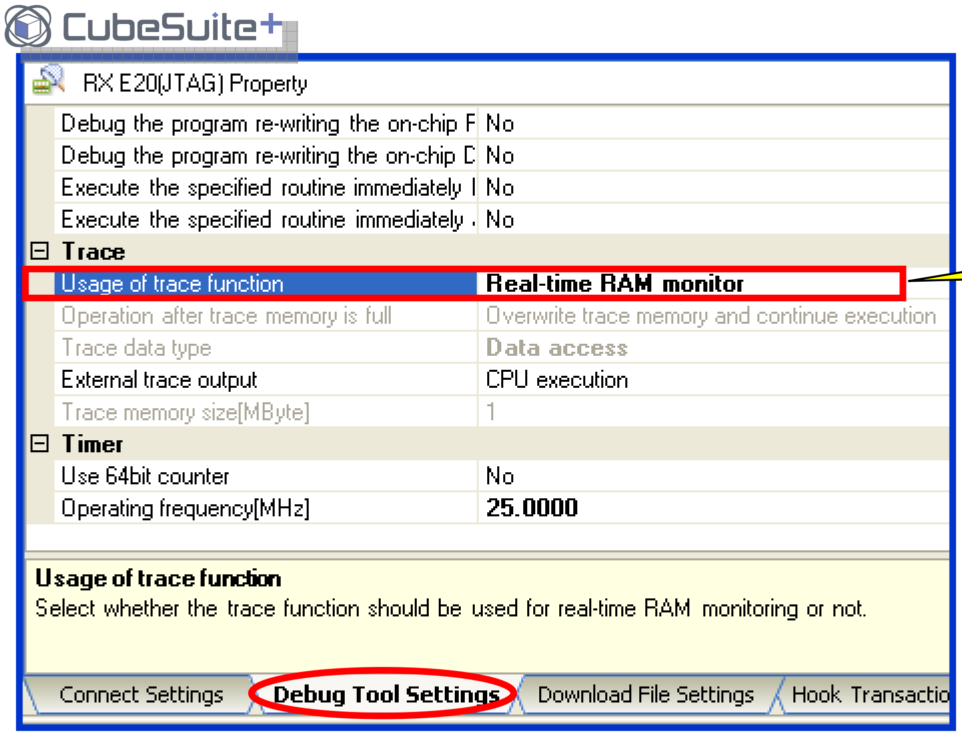
Specify a memory range.

Note: Enter decimal numbers here. When entering hexadecimal numbers, add “0x” to the head of each number.

23. RAM Monitoring (When the E20 is Used)

CubeSuite+ provides the RRM (Real-time RAM Monitor) function that is equivalent to the HEW's RAM monitoring. The following describes how to use the RRM function:

23.1 Set [Usage of trace function] in the [Debug Tool Settings] sheet on the [Properties] panel of the debug tool to [Real-time RAM monitor].



RX E20(JTAG) Property

Debug the program re-writing the on-chip F	No
Debug the program re-writing the on-chip D	No
Execute the specified routine immediately I	No
Execute the specified routine immediately .	No
⊟ **Trace**	
Usage of trace function	**Real-time RAM monitor**
Operation after trace memory is full	Overwrite trace memory and continue execution
Trace data type	**Data access**
External trace output	CPU execution
Trace memory size[MByte]	1
⊟ **Timer**	
Use 64bit counter	No
Operating frequency[MHz]	**25.0000**

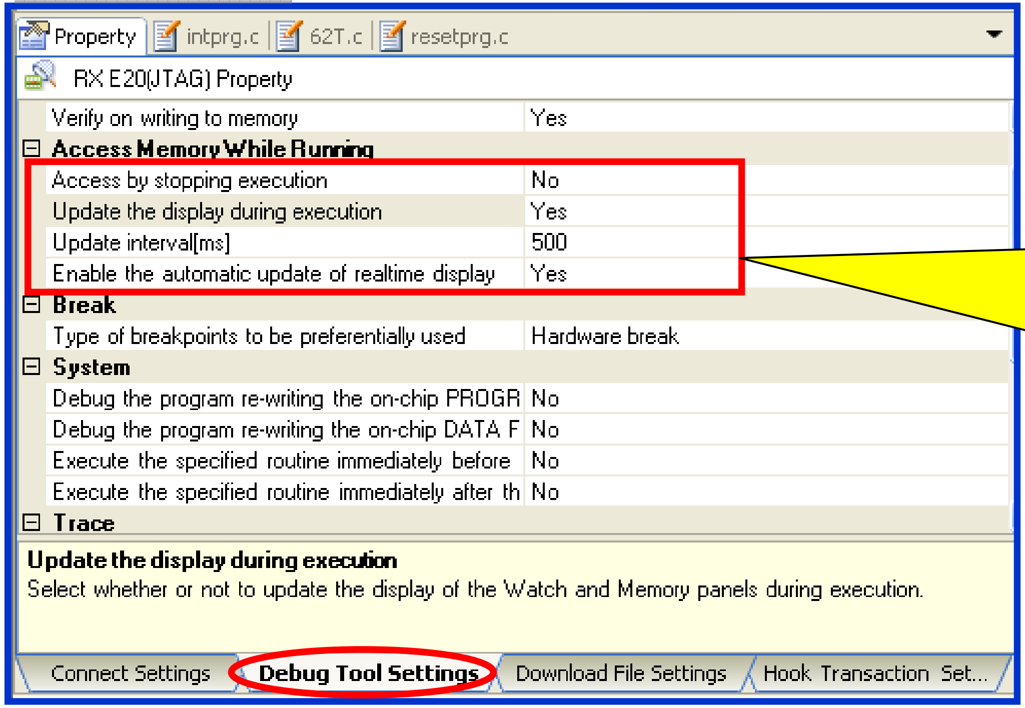
Usage of trace function
Select whether the trace function should be used for real-time RAM monitoring or not.

Set Usage of trace function to [Real-time RAM monitor].

23. RAM Monitoring (When the E20 is Used)

23.2 In the [Debug Tool Settings] sheet, set [Access by stopping execution] to [No], [Update the display during execution] to [Yes], and [Enable the automatic update of realtime display] to [Yes].
Set a desired update interval for [Update interval].

CubeSuite+



RX E20(JTAG) Property

Verify on writing to memory	Yes
⊟ **Access Memory While Running**	
Access by stopping execution	No
Update the display during execution	Yes
Update interval[ms]	500
Enable the automatic update of realtime display	Yes
⊟ **Break**	
Type of breakpoints to be preferentially used	Hardware break
⊟ **System**	
Debug the program re-writing the on-chip PROGR	No
Debug the program re-writing the on-chip DATA F	No
Execute the specified routine immediately before	No
Execute the specified routine immediately after th	No
⊟ **Trace**	

Update the display during execution
Select whether or not to update the display of the Watch and Memory panels during execution.

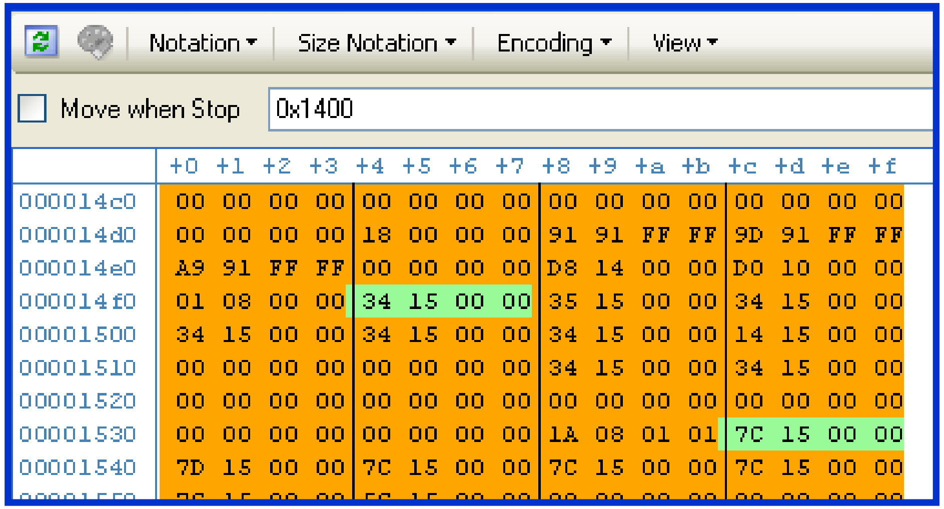
Make the following settings:
[Access by stopping execution]: No
[Update the display during execution]: Yes
[Update interval]: Desired update interval
[Enable the automatic update of realtime display]: Yes

Note: If [Access by stopping execution] is set to [Yes], the RRM function is not usable.

23. RAM Monitoring (When the E20 is Used)

23.3 Open the area you want to view with real-time RAM monitoring on the [Memory] panel, and then execute the program.
Memory data is displayed by the RRM function.

CubeSuite+



☐ Move when Stop   0x1400

```
          +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f
000014c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000014d0  00 00 00 00 18 00 00 00 91 91 FF FF 9D 91 FF FF
000014e0  A9 91 FF FF 00 00 00 00 D8 14 00 00 D0 10 00 00
000014f0  01 08 00 00 34 15 00 00 35 15 00 00 34 15 00 00
00001500  34 15 00 00 34 15 00 00 34 15 00 00 14 15 00 00
00001510  00 00 00 00 00 00 00 00 34 15 00 00 34 15 00 00
00001520  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00001530  00 00 00 00 00 00 00 00 1A 08 01 01 7C 15 00 00
00001540  7D 15 00 00 7C 15 00 00 7C 15 00 00 7C 15 00 00
```

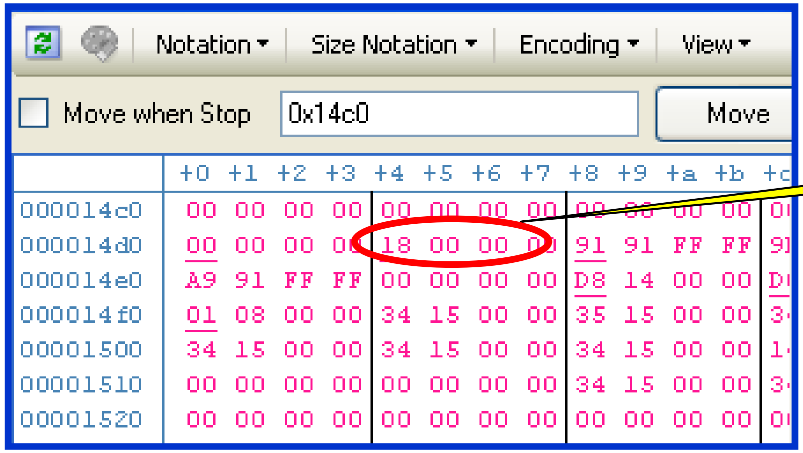
Example of display by the RRM function

00	: Read/fetch
00	: Write
00	: Read/write
00	: No access
00	: Loss of data
**	: An area other than the real-time display update area was specified while the program is running or acquisition of memory data failed

23. RAM Monitoring (When the E20 is Used)

23.4 Troubleshooting

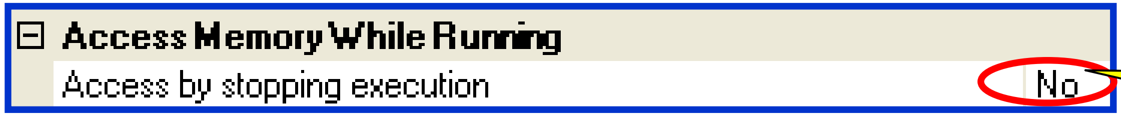
(1) Values on the [Memory] panel are updated, but the background color remains white.



```
          +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c
000014c0  00 00 00 00 00 00 00 00 00 00 00 00 0
000014d0  00 00 00 00 18 00 00 00 91 91 FF FF 9
000014e0  A9 91 FF FF 00 00 00 00 D8 14 00 00 D
000014f0  01 08 00 00 34 15 00 00 35 15 00 00 3
00001500  34 15 00 00 34 15 00 00 34 15 00 00 1
00001510  00 00 00 00 00 00 00 00 34 15 00 00 3
00001520  00 00 00 00 00 00 00 00 00 00 00 00 0
```

Values are updated, but background color remains white.

The RRM function is disabled.
If [Access by stopping execution] in the [Debug Tool Settings] sheet on the [Properties] panel is set to [Yes], the RRM function is not usable.
Set [Access by stopping execution] to [No].



⊟ **Access Memory While Running**

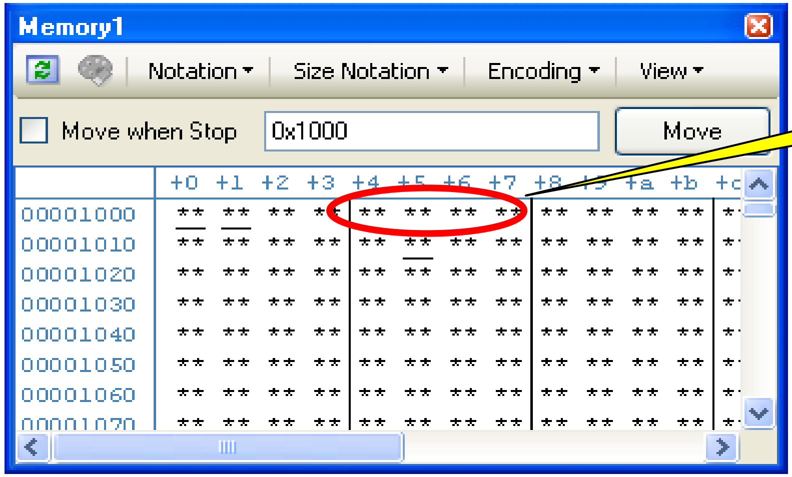
Access by stopping execution No

Set this item to [No] if the RRM function is to be used.

23. RAM Monitoring (When the E20 is Used)

23.4 Troubleshooting

(2) “**” is displayed on the [Memory] panel.



□ Move when Stop 0x1000

```
         +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c
00001000 ** ** ** ** ** ** ** ** ** ** ** ** *
00001010 ** ** ** ** ** ** ** ** ** ** ** ** *
00001020 ** ** ** ** ** ** ** ** ** ** ** ** *
00001030 ** ** ** ** ** ** ** ** ** ** ** ** *
00001040 ** ** ** ** ** ** ** ** ** ** ** ** *
00001050 ** ** ** ** ** ** ** ** ** ** ** ** *
00001060 ** ** ** ** ** ** ** ** ** ** ** ** *
00001070 ** ** ** ** ** ** ** ** ** ** ** ** *
```

“**” is displayed.

The displayed area is not covered by the RRM function.

The RRM function covers an area of up to 4096 bytes (1024 bytes × 4) that is automatically allocated according to the data on the open [Watch] and [Memory] panels.

If the amount of data displayed on the [Memory] panel is larger than the area allocated for the RRM function, the RRM function is not usable.

In that case, perform the following:

(a) Reduce the number of variables registered in the [Watch] panel or close the [Watch] panel.

(b) Open only the [Memory] panel on which you want to use the RRM function.

Renesas Electronics Corporation