CubeSuite+ V1.03.00

Release Note

R20UT2259EJ0100
Rev. 1.00
October 1, 2012

Chapter 1. Target Devices

The target devices supported by the CubeSuite+ are listed on the Website.
Please see this URL.
CubeSuite+ Product Page:
http://www.renesas.com/cubesuite+

Chapter 2. User's Manuals

Please read the following user's manuals together with this document.

Manual Name	Document Number
CubeSuite+ V1.03.00 Start	R20UT2133EJ0100
CubeSuite+ V1.03.00 78K0 Design	R20UT2138JJ0100
CubeSuite+ V1.03.00 78K0R Design	R20UT2137JJ0100
CubeSuite+ V1.03.00 RL78 Design	R20UT2136JJ0100
CubeSuite+ V1.03.00 V850 Design	R20UT2134JJ0100
CubeSuite+ V1.01.00 78K0 Debug	R20UT0731EJ0100
CubeSuite+ V1.01.00 78K0R Debug	R20UT0732EJ0100
CubeSuite+ V1.03.00 RL78 Debug	R20UT2145JJ0100
CubeSuite+ V1.03.00 V850 Debug	R20UT2144EJ0100
CubeSuite+ V1.03.00 RX Debug	R20UT2175EJ0100
CubeSuite+ V1.03.00 Analysis	R20UT2146EJ0100
CubeSuite+ V1.03.00 Message	R20UT2147EJ0100

Chapter 3. Key Points for Selecting Uninstallation Method

There are two ways to uninstall this product.

- Use the integrated uninstaller (uninstalls CubeSuite+)
- Use separate uninstaller (uninstalls this product only)

To use the separate uninstaller, select the following from the Control Panel:

- Add/Remove Programs (Windows XP)
- Programs and Features (Windows Vista / Windows 7)

Then select "CubeSuite+".

Chapter 4. Changes

This chapter describes changes from V1.02.00 to V1.02.01

4.1 Improvement of CubeSuite+ functionality

4.1.1 Addition of functions for changing the microcontroller

The function enables users to change a specified microcontroller with another microcontroller of the same device family.
microcontroller to another from the same family. However, the specified build tool for the project isn't allowed to change.
Disconnect the debug tool, and select [Change Microcontroller…] from the context menu in the node showing the microcontroller name in the Project Tree panel.

4.1.2 Addition of display functions

A function for zooming in and out on the display in each panel was added . Regarding the panels and how to zoom the displays in and out, refer to the online help or the user's manual.
The ability to import and export the settings for color specified in the Option dialog box was also added.

4.1.3 Addition of functions to the Project Tree panel

A function for indicating dependency files due to #include directives in the source files in the Project Tree panel was added.
A function to bring up the context menu of Windows Explorer was added to allow easy access to and use of other tools (e.g., the source management tool) that have been installed under Windows was also added.

4.1.4 Addition of functions for the Property panel

A function for storing the tabbed pages displayed for each tool in the Property panel was added. Each time the user switches to the property panel, the tabbed page that was previously being displayed is shown rather than the top tabbed page.

4.1.5 Improved functionality for the Output panel

The display in the Rapid Build tabbed page of the Output panel has been changed and is now a cumulative display. This enables the checking of source files built as rapid builds and of messages in builds even if multiple source files have been edited or saved.

4.1.6 Addition of functions for plug-in management

Functions for the control of plug-ins of the build tool and the debug tool were added to the plug-in management functions.

4.1.7 Improvement of a converting function from the High-performance Embedded Workshop into CubeSuite+

A converting function High-performance Embedded Workshop projects into CubeSuite+ is improved. The function converts folders for the High-performance Embedded Workshop project tree into CubeSuite+ categories.

4.1.8 Improved functionality for the online help system

The tutorial display function only displays the tutorial guide portion of the online help. The help display function, as in the previous release, displays both the contents of the tutorial and of the online help itself. Furthermore, descriptions of messages were also added to the online help that is displayed by the update manager.

4.1.9 Improved performance

Performance in processing for the reading of projects and the performance of the editor have been improved.

4.1.10 Improved update feature

Network environment which requires proxy authentication has been supported. And resume function has been supported.

4.2 Improvements to the editor

4.2.1 Addition of smart edit functions

A smart edit function, which displays candidates being input, has been added (for the RX and V850E2 families).

4.2.2 Improved dragging and dropping functions

The ability to drag and drop variables to register them for the program analysis tool was added to the Variable Value Changing Chart tab of the Analysis Chart panel.

4.2.3 Improvement of print functions

The abilities to include line-numbers and to handle information from mixed display mode were added to the print and print preview functions.

4.2.4 Addition of column headers

Column headers were added under the toolbar to make it easier to see the functions of each area.

4.2.5 Addition of control over encoding

The ability to enable or disable the automatic determination of encoding was added. This enables the opening of files by using the default encoding if incorrect recognition occurs in the automatic determination of encoding.

4.3 Improvements to the debugging tool

4.3.1 Addition of the ability to reset events

The ability to reset action events was added.
The ability to reset specifications of break conditions regarding access to variables was added (RL78 and 78K0R).
Select the event to be reset in the Event panel and select [Edit Condition...] from the context menu.

4.3.2 Addition of a property (RX)

Modifications were made to save the settings for power supply in the project. Also, the ability to specify the erasure of flash ROM at the time of downloading for both code flash and data flash memory was added.

4.3.3 Addition of the Output panel (RX)

Information on the E1 or E20 such as its version number and the voltage of the user system is displayed in the Output panel when an E1 or E20 emulator for RX devices is connected..

4.3.4 Addition of the MPU area access (RX)

The ability to access the MPU area of CPU was added.

4.3.5 Additions to the Watch panel

The ability to import and export the settings being monitored on the watch panel was added.
To import settings of watch panel symbols, select [Import Watch Expression] and specify the file to be imported. To export the current settings, select the [Save Watch Data] menu item under [File], or the [Save Watch Data As...] menu item under [File], and select "import-possible CSV (comma-delimited)" for the file type and then save the settings in a file.

4.3.6 Additions to the Memory panel

The ability to specify the number of digits of addresses to be displayed, the address where the display starts, and the range of addresses displayed on the memory panel was added.

4.3.7 Correction of Python function in standby mode

Applies to: Simulator, 78K0 / 78K0R / RL78 / V850 / V850E2
The return value of Python function, debugger.GetCpuStatus(), in standby mode such as STOP or HALT has been corrected.

4.4 Improvement of the program analysis tool

4.4.1 Addition of property functions

Among the properties of the program analysis tool, the ability to specify the enabling of static or dynamic analysis has been added. Note that detailed settings are specified by a build tool and debugging tool.

4.4.2 Addition of functions to the Variable Value Changing Chart tab

The ability to use a mouse to modify offsets on graphs was added. An offset can be modified by placing the mouse cursor on the point to be changed while pressing the [Shift] key and then dragging and dropping. A function for automatically adjusting graphs was also added. Double-clicking on "Val/Div" in the area where variables for the explanatory area are registered displays the graph of the relevant channel so that the upper limit of the displayed range corresponds to the maximum value of the data for the graph, while its lower limit corresponds to the minimum value of the data. The graph is thus expanded to fill the entire drawing area.

4.5 Improvement of Python console

4.5.1 Addition of Python functions and Python property

Following Python functions and Python property are added. Please refer "Appendix G" in a user's manual "CubeSuite+ V1.03.00 Start"
Additional Python functions : project.Create, common.OutputPanel
Additional Python property : debugger.ADConvertDataInExecution

4.5.2 Addition of return values of Python functions

The return values of following Python functions are added. We can see whether functions are performed successfully or not from these return values (True or False).

Python function for project	Save
Python function for build tool	build.ChangeBuildMode
Python function for debug tool	debugger.Assemble.LineAssemble, debugger.Breakpoint.Enable, debugger.Breakpoint.Disable, debugger.Breakpoint.Delete, debugger.Connect, debugger.DebugTool.Change, debugger.Disconnect, debugger.Download.Binary, debugger.Download.Binary64Kb, debugger.Download.BinaryBank, debugger.Download.Hex, debugger.Download.Hex64Kb, debugger.Download.HexBank, debugger.Download.HexIdTag, debugger.Download.Coverage, debugger.Download.LoadModule, debugger.Erase, debugger.Ie.SetValue, debugger.Map.Set, debugger.Map.Clear, debugger.Memory.Write, debugger.Memory.Fill, debugger.Memory.Copy, debugger.Register.SetValue, debugger.Upload.Binary, debugger.Upload.Coverage, debugger.Upload.Intel, debugger.Upload.IntelIdTag, debugger.Upload.Motorola, debugger.Upload.MotorolaIdTag, debugger.Upload.Tektronix, debugger.Upload.TektronixIdTag, debugger.Watch.SetValue, debugger.XCoverage.Clear, debugger.XRunBreak.Delete, debugger.XRunBreak.Set, debugger.XTrace.Clear

4.6 Improvement of Stack Usage Tracer

4.6.1 Improvement of Stack Usage Tracer for V850E2

Following have been improved.
- Indication of the function which contains an indirect function call.
- Support of subroutinization.

4.7 Improvement of Emulator utilities

4.7.1 Improvement of E1,E20 Self Check Program

The processing performance when E1,E20 Self Check Program is started has been improved.

4.8 Removal of cautions

4.8.1 Caution on memory panel while using RRM function

Applies to: E20 Emulator(JTAG),RX

If the value of the variable that was set in the watch panel by using RRM function is displayed in a memory panel , the memory panel displays not "**" but "00".

Use Watch panel instead of Memory panel.

4.8.2 Cautions on using the RX (simulator and E1/E20 emulator)

debugger.Assemble.LineAssemble() function does not support Big Endian, so this function dose not operate correctly when using it for Big Endian area.

4.8.3 Cautions on using "CubeSuiteExit()" function and Rapid start function

When closing CubeSuite+ by using "CubeSuiteExit()" function, CubeSuite+ does not become a resident program even if Rapid start function was used.

4.9 Removal of restriction

4.9.1 Removal of restriction for setting breakpoints

Restriction below has been removed.

- In the systems development using any MCU of the RL78 family or 78K0R whose ROM capacity is equal to or greater than 96 KB, if you set a breakpoint by using either of the above simulators, the data in a specific address other than the breakpoint may be changed to 0xFF.

Chapter 5. Cautions

This section describes cautions for CubeSuite+.

5.1 Cautions for CubeSuite+ (general)

5.1.1 Cautions for file names

The following cautions apply to folder and file names.

- Folder and file names
  Do not use folder or file names that cannot be created from Windows Explorer.
- Source file names, load module file names, and project file names
  File names consist of the characters a-z, A-Z, 0-9, period (.), underscore (_), plus (+), and minus (-).
  File names cannot end with a period (.).
  File names cannot start with a period (.).
  Filenames cannot start with a plus sign (+) or minus sign (-).
  File names are case-insensitive.
  File names may be up to 259 characters, including the path.
- File names other than the above.
  File names comply with Windows conventions.
  Note that the following characters cannot be used in file names.
  \ / : * ? " < > | ;
  File names cannot start with a period (.) or space.
  The uppercase and lowercase characters of the file name are not distinguished.
  File names may be up to 259 characters, including the path.
- Folder names
  Comply with Windows file name conventions.
  Note that the following characters cannot be used in file names. (Excluding RL78, 78K0, 78K0R, and V850 projects) ( ) , =

5.1.2 Caution for panel displays

If your hardware environment does not meet the recommended specifications for CubeSuite+, the [Property] panel may appear small, and the contents scrambled.
If this happens, move the [Property] panel outside the split panel area.

- Enable Dockable, and make it a Docking panel
- Enable Floating, and make it a Floating panel

5.1.3 Caution for User Account Control (UAC) function (Windows Vista)

If the UAC function is disabled on Windows Vista / Windows 7, then if a user without administrator privileges creates a project, and no Device Dependence Information is installed, then the installation of the Device Dependence Information will begin, but the installation will fail. If the UAC function is disabled, create projects after logging in with administrator privileges.

5.1.4 Caution for command accelerators included in split panels/categories

Although accelerators are displayed in the command menus of split panels and categories, pressing the keyboard shortcuts will have no effect. Use the mouse to select menu items.

5.1.5 Caution for Windows update program

Your computer may suffer "blue screen" errors if you apply the KB2393802 patch published by Microsoft Corporation. If this error occurs, please apply the patch provided by your computer's manufacturer or other source.

5.1.6 Caution for Renesas Electronics real-time OS

If you use the real-time operation system for the RX family provided by Renesas Electronics, install CubeSuite+ to a folder path that does not contain parentheses. If you install it to the 64-bit version of Windows, it will be installed in the "Program Files (x86)" folder by default, and if the folder path includes parenthesis characters, it will result in an error.

5.1.7 Caution with regard to changing the microcontroller

Note the following points for caution when changing the microcontroller.

- The microcontroller can only be changed to another within the same family, since this will correspond to the same build tool (V850, RX, RL78, 78K0R, and 78K0).
- When changing the microcontroller, do so while the debugger is not connected.
- Save the project before changing the microcontroller.
- Information on pin layout (design tool), code generation (design tool), and debugging (except for watch registration information) are not carried over after the microcontroller has been changed.

5.1.8 Caution with the Plug-in Manager Function

We recommend that the checkbox for the plug-in for the microcontroller that is the target for development is not deselected on the Basic Function tabbed page of the Plug-in Manager dialog box.
Deselect the checkboxes for the build tool and debugging tool plug-ins that are for microcontrollers that are not the target for development.For example, if only the plug-in for the build tool is deselected, the file to be downloaded by the debugging tool will not be found and an error will occur.

5.1.9 Cautions for Editor panel

- When you change the active file using the File tab, the "Forward to Next Cursor Position" and "Back to Last Cursor Position" features may not work.
- The "Show whitespace marks" feature does not display spaces in two-byte encoding.
- The Page Setup dialog box is not available.
- Although there is a Copy button on the Print Preview toolbar, it does not work..
- When a variable or label is selected and the Jump to Function feature is used from the context menu, execution does not jump to the variable or label.
- The Jump to Function feature will not jump to a static function defined in another file.
- The following cautions apply to editor, when the source file of the same name from which a folder is different was registered with a main project and a sub project, and downloading a load module of a main project and a sub project both.
  - The address of the main project is displayed on the file.
  - At jumping to a source file from disassembly code, the file registered with a main project is opened.
  - If the file is opened from whichever project s, only 1 file is opened.
- The smart edit function does not run correctly for a union or structure that has been defined by using a typedef declaration.
- The smart edit function does not run correctly for a structure that does not have a name.
- If the arguments of a function include a function call, incorrect information will be displayed in the tooltip.
- Member variables or functions are not correctly complemented in arrays of classes and of pointers to classes.
- Complementation does not produce appropriate strings even if a part of a member name is entered

and 'ctrl + (space)' is also entered.

- The following keywords for C++ are also highlighted in C files. They are only highlighted for display and do not affect other processing.
  - When the build tool is CC-RX
    __cplusplus、and、and_eq、bitand、bitor、catch、class、compl、const_cast、delete、dynamic_cast、explicit、friend、mutable、namespace、new、not、not_eq、operator、or、or_eq、private、protected、public、reinterpret_cast、static_cast、template、this、throw、try、typeid、typename、using、virtual、wchar_t、xor、xor_eq
  - When the build tool is CA78K0, CA78K0R, CA850, or CX
    bool, w_char, true, false

5.1.10 Caution for conversion from PM+ to CubeSuite+ project.

CubeSuite+ can't read the CA850 project, if the project is made by PM+ V6.00/V6.10/V6.11 and is added a new Build Mode. It'll be as follows.

1)When Debug Build or Release Build is specified, information on the added Build Mode can't be read.
2)When the added Build Mode is specified, it'll be an error.

[Workaround]
Please read the project by more than PM+ V6.20 and save it.

5.1.11 Cautions for Debugging Tool Settings during Project Composition

When you create a project by composite the settings of another project, only the settings for the default debugging tool will be imported. In the RX family, however, internal processing is common to the emulator and simulator, so the settings are imported regardless of which debugging tool is selected.

5.1.12 Cautions for Online Help

If you close the online help while the Search tab is displayed, and you then display the online help again and display the Contents tab, the Coding and Build editions may disappear.
If this happens, close the online help with the Contents tab displayed, and then display the online help again.

5.1.13 Cautions for project conversion

When changed conversion device of a project in [project convert setting] dialog when opened High-performance Embedded Workshop/PM+/CubeSuite, return a value chosen in [king of project] to the value that an initial value appears.

For example:
A kind of a project is replaced by the top (for example, [Application]) when chooses a device again in the back.

5.1.14 Note on Conversion of High-performance Embedded Workshop Projects

If you attempt to load a High-performance Embedded Workshop project into the CubeSuite+ under certain conditions, an error may occur during conversion or building of the project.

(1) Converting a High-performance Embedded Workshop project to make it compatible with the CubeSuite+ fails when any of the following conditions is satisfied.

- No toolchain from Renesas Electronics Corp. is selected for the project.
- The project contains no tps file that is used to set up the High-performance Embedded Workshop environment. (the tps file is automatically created when the project is opened through the High-performance Embedded Workshop). To avoid this problem, you should open the project through the High-performance Embedded Workshop once before starting conversion.
- The project contains multiple CFG files, each of which is used to set up the realtime OS from

Renesas Electronics Corp.

(2) Converting a High-performance Embedded Workshop project to make it compatible with the CubeSuite+ succeeds but building of the project leads to an error when any of the following conditions is satisfied.

- Placeholder $(TCINSTALL) is used in the project.
  $(TCINSTALL) remains in the project even after conversion but the CubeSuite+ does not recognize $(TCINSTALL). Placeholder $(TCINSTALL) that has been used as a parameter for [Options] in the High-performance Embedded Workshop is simply passed to the CubeSuite+ and may cause an unintended result (e.g. an error) upon building of the project. For this reason, you should manually change $(TCINSTALL) after converting the project.
- Placeholder $(WORKSPDIR) is used in the project.
  If you select a HEW project file (with extension hwp) in the CubeSuite+, it is automatically converted to “%ProjectDir%\..” (the directory above the project directory). An error may occur during building of the project if the workspace does not exist in the directory indicated by “%ProjectDir%\..”.
  For this reason, you should manually change “%ProjectDir%\..” after converting the project.
- A custom build phase is used in the project.
  Since all custom phases are deleted upon conversion, an error may occur during building of the project that involved a file output created for a custom build phase in the High-performance Embedded Workshop.
  After converting the project, register the custom build-phase command with the CubeSuite+ as a command to be executed before or after each phase as required.
- A custom placeholder is used in the project.
  Custom placeholders are not converted because the CubeSuite+ does not recognize them. Any custom placeholder that has been used as a parameter for [Options] in the High-performance Embedded Workshop is simply passed to the CubeSuite+ and may cause an unintended result (e.g. an error) upon building of the project. For this reason, you should manually change the custom placeholder after converting the project.

(3) Other

(a) $(FILEDIR) is converted to %FileDir%.
Leaving %FileDir% as it is when the pathname is edited in the [Path Edit] dialog box will lead to the following error: The specified path contains a non-existent folder. (W0205012)
When you edit the pathname, replace %FileDir% with another placeholder or directory name.

(b) $(WINDIR) is converted to %WinDir%.

(c) The order in which folders are displayed in CubeSuite+ may differ from that in the High-performance Embedded Workshop.

(d) If a High-performance Embedded Workshop project for which downloaded files have been specified is loaded into CubeSuite+, CubeSuite+ will show these files as the second and subsequent items in the list of downloaded files for each debug tool.

(e) The compiler option -output=src is converted to -output=obj (default).

(f) If you load a library project that has been linked to the standard library into CubeSuite+, the linkage setting will be discarded (this is indicated in the log information that is output as a result of loading the project).

(g) If [Use an existing library file] has been selected for the library generator in the High-performance Embedded Workshop, the setting is changed to [Do not add a library file] in CubeSuite+. For this reason, linking with the specified library will not proceed (this is indicated in the log information that is output as a result of loading the project).

(h) Option settings that were made on the [Toolchain Option] tabbed page of the High-performance Embedded Workshop are not converted but discarded (i.e. they are not moved across to CubeSuite+).

(i) If a sub-command file has been selected for the linkage editor in the High-performance Embedded Workshop, the [Use external subcommand file] setting is discarded when the project is loaded into CubeSuite+. The linkage editor options will have their default settings.

(j) Any files specified with the -library, -input, or -binary option will not be listed in the [Link Order] dialog box. The result is that the order of linkage for these files will not be selectable.

(k) RTOS configuration files will not be displayed under the [Configuration file] category node after the project is loaded into CubeSuite+.

(l) RTOS option settings that were made in the High-performance Embedded Workshop are discarded. RTOS options will have their default settings in CubeSuite+.

(m) The build mode for RTOS projects will be "DefaultBuild" after the project is loaded into CubeSuite+. You will need to change the build mode as required.

(n) The order of linkage of the assembly output file (ritbl.obj) in an RTOS project will differ from that in the High-performance Embedded Workshop.

5.1.15 Caution for creating new projects

Applies to: RX

If a new project is created by selecting [Empty Application[CC-RX]] under the environment for the RX, building the project may lead to the following errors.

** L2132 (E) Cannot find "D" specified in option "rom"
** L2132 (E) Cannot find "D_1" specified in option "rom"
** L2132 (E) Cannot find "D_2" specified in option "rom"

If you encounter these errors, change the setting of [ROM to RAM mapped section] on the [Link Options] sheet in CubeSuite+.

5.2 Cautions for Design Tool

5.2.1 Caution for changing packages

If you change the package name in the pin layout properties, the data input in the device top view and device pin list will be cleared.

5.2.2 Caution for saving projects

If you save a project that has sub-projects while the Device Top View or Device Pin List panel is open, then the device top view and device pin list of the last sub-project in the Project Tree will always appear.

5.2.3 Caution for saving projects

Applies to: 78K0 / 78K0R / RL78

A value of “Use on-chip debug” or “Set user option byte” on [Link Option] Tab may be different between on saving a project file and on reading a project file.

[Condition]

1) The log-in user’s own .mtud file doesn’t exist when reading a project file
   Example 1
   Another log-in user B read the project file that a log-in user A saved
   Example 2
   A log-in user A read a project after the user saved the project file and deleted .mtud file by intent.

2) The log-in user’s own .mtud file exists and the cord generation panel is displayed on the forefront.

[Procedure]

After "read project file" or before "build", verify that the values are correct.

5.3 Cautions for Build Tool

5.3.1 Cautions for startup node

In the case of a CX project, the following warning will be output if an object module file (.obj) is registered to the startup node. Please ignore this warning.

W0560111: The same file is specified multiple times as an input file.

In the case of a multicore project, the following error will be output if an object module file (.obj) is registered to the startup node. Please register an assembler file (.asm) to the startup node.

F0560208: Symbol “xxx” has been defined more than once.

5.4 Cautions for debugging tool

The below abbreviated names are used in this section.

OCD(Serial)	: MINICUBE2, E1 Emulator(Serial), E20 Emulator(Serial)
OCD(JTAG)	: MINICUBE, E1 Emulator (JTAG), E20 Emulator (JTAG)

5.4.1 Caution for adding sub-projects

Applies to: All debugging tools,Common to all devices

Disconnect the debugging tool before adding a sub-project that handles a different device than the main project.

5.4.2 Caution for executing a boot swap

Applies to: Simulator/ OCD(JTAG)/ OCD(Serial), V850 / 78K0 / 78K0R / RL78

If a software break is set in a boot-swap area, then a break instruction will be written to the Flash ROM. For this reason, a break instruction will remain after the boot swap.

- OCD(JTAG)/ OCD(Serial) : Use a hardware break if you wish to set a breakpoint.
- Simulator : Don't use break point in this area.

5.4.3 Caution for executing programs in internal RAM area

Applies to: Simulator, V850

The following cautions apply when executing programs within the internal RAM (addresses from 0x0fff0000 to 0x0fffefff) of the V850E/MA3 and other V850E microcontrollers.

- When CPU stops, the displayed address on disassemble panel is "0x03ff0000 to 0x03ffefff".
- In the case using "Step-In" execution for the function in internal RAM, the actual operation become "Step -Over".

5.4.4 Caution for standby mode

Applies to: All debugging tools, V850 / 78K0 / 78K0R / RL78

If a forced break is performed while in standby mode (e.g. STOP mode or HALT mode), or an instruction to move to standby mode is made while in step execution, then behavior will differ between the simulator and the emulator (IECUBE, OCD(JTAG), and OCD(Serial)).

- Emulator: The forced break will release standby mode. In step execution, it will not go into standby mode.
- Simulator: The forced break will not release standby mode. In step execution, it will go into standby mode.

In either case, the program counter (PC) row upon forced break will break at the next instruction after the standby mode instruction (e.g. HALT). Thus in the case of the simulator, it will appear that standby mode has been released. Check the status bar to see if standby mode has been released. If the simulator is in standby mode, "Halt" or "Standby" will appear in the status bar.

5.4.5 Caution on low-power consumption modes

Applies to: All debugging tools, RX

When a forced break occurs in a low-power consumption mode (e.g. sleep, stop, or standby) or an instruction that makes the CPU enter a low-power consumption mode is executed during stepped execution, the behavior of the simulator and the emulator will differ as follows.

- Emulator: A forced break releases the CPU from the low-power consumption mode. On the other hand, the CPU can enter a low-power consumption mode during stepped execution.
- Simulator: Transition to a low-power consumption mode (e.g. by a register setting) is not supported. Executing a WAIT instruction causes a break, with the PC placed at the address of the next instruction. During stepped execution, the CPU also does not enter a low-power consumption mode and the PC is placed at the address of the next instruction.

5.4.6 Caution for multipliers/dividers

Applies to: Simulator, 78K0

When simulating 78K0 instructions, the multiplier and divider are not supported. For this reason, to perform multiplication or division within the program, from the build tool, open the Property panel, and on the [Compiler Options] tab, from the [Use multiplier/divider] drop-down list, select [No].

5.4.7 Caution for Memory bank function

Applies to: Simulator, 78K0

When simulating 78K0 by instruction mode, the memory bank function is not supported.

5.4.8 Caution for CPU operation clock

Applies to: Simulator, 78K0R / RL78

- When simulating 78K0R by instruction mode, the frequency of on chip oscillator is 8MHz.
- When simulating RL78 by instruction mode, the CPU operation clock operates by the specification of RL78/G13

5.4.9 Caution for Multiplier and Divider/Multiply-Accumulator

Applies to: Simulator, 78K0R / RL78

When simulating 78K0R or RL78 by instruction mode, cautions of Multiplier and Divider/Multiply-Accumulator are following.

(1) When using it by division mode, the division processing will be finished in by 1 clock.

(2) When using it by division mode, the interrupt "INTMD" (the end of division operation) is not occurred. But DIVST bit of Multiplication/Division Control Register "MDUC" is changed. (DIVST bit displays division operation status.)

5.4.10 Caution for traces in arbitrary intervals

Applies to: Simulator, all devices

If you perform a trace from a trace start event until a trace end event, the simulator will not display the trace end event as the results of the trace. For this reason, if you are using a simulator, set the trace end event to one line below the range that you wish to display as the trace data.

5.4.11 Caution for runtime measurement over arbitrary intervals

Applies to: Simulator, V850 / 78K0 / 78K0R / RL78

If you measure run time from a timer start event until a timer end event, the simulator will not include the time for the timer end event in the measurement results. For this reason, if you are using a simulator, set the timer end event to one line below the range for which you wish to measure the run time.

5.4.12 Caution for CPU operation clock

Applies to: Simulator, V850

The clock generator is not simulated in V850 instruction simulation mode. For this reason, the CPU operation clock will always have the main clock frequency set in the Properties panel (even if a built-in peripheral I/O register with a clock generator is manipulated, the CPU's operation clock will not change)

5.4.13 Caution for displaying maximum address space in memory display panel

Applies to: OCD(Serial) / IECUBE, 78K0

To access the device maximum internal ROM, internal fast RAM, or internal extended RAM sizes from the memory panel or the like, set a hook process in the memory size switch register (IMS) and internal extended RAM size switch register (IXS).

5.4.14 Caution for retuning execution and displaying the call stack

Applies to: All debug tools, 78K0R / RL78

If step execution is performed from the editor panel (in source mode), the debugging tool determines whether an interrupt is being processed via the NP, EP, and ID flags in the POWER switch register. For this reason, if the above flags or register are changed (e.g. when using multiple interrupts), then the return execution and displaying the call stack may be incorrect.

5.4.15 Software breakpoints when ROMification has been performed

Applies to: All debug tool, RL78 / V850

If there is code (text sections) to be ROMified, any breakpoint instructions set for that code will be deleted during rcopy. For this reason, no break will occur. Use a hardware break if you are using OCD(JTAG) or OCD(Serial) or IECUBE. Note that if you are using a simulator, execution will not break even if a hardware breakpoint is used, but it will break if the tracer or timer is turned on.

5.4.16 Adding sub-projects

[Applies to] Common to all debug tools and devices

If you add a sub-project while a debugging tool is connected, downloading and the like may fail. Add sub-projects while the debugging tool is disconnected.

5.4.17 Configuring Flash options

[Applies to] OCD(JTAG), V850E2M

The bit shown below indicating the following Flash option has been locked to 1. Use a Flash programmer

if you wish to write 0 to it.
- Bit 95 of on-chip debugging security ID (security lock signal release)
- Bit 31 of option byte 0 (debug interface connection disabled bit)

5.4.18 Caution for Stack-trace display

[Applies to] All debugging tools, 78K0

The stack-frame display function may fail to correctly display up to the **main** function if a function is used that does not push the frame pointer (HL) onto the stack (e.g. **noauto** or **norec** function), or if the memory bank is used.

Additionally, a free-run state may occur if a return is executed from a function that does not push the frame pointer (HL) onto the stack (e.g. **noauto** or **norec** function), or if a memory bank function is used.

5.4.19 Caution for stepping into main bank

[Applies to] All debugging tools, 78K0

If you step into a user-defined library function or function without debugging information in the memory bank at the source level, execution will break in the bank-switching library.

5.4.20 Caution for local-variable display

[Applies to] All debugging tools, 78K0

Local variables outside the scope of the current PC are not displayed correctly in the stack trace panel.

5.4.21 Caution for disassemble window

[Applies to] All debugging tools, 78K0

When displaying instructions in the common area in the disassemble window, if the displayed instruction uses a symbol in the memory bank area, a symbol from a different bank may be displayed.

5.4.22 Breakpoint and other settings become invalid

[Applies to] Common to all debugging tools and devices

If you differentiate function or variable names by leading underscores, then the debugger may misrecognize them, and convert symbols or make breakpoint settings invalid.

This applies for cases like when you have two functions, one named _reset and the other named __reset.

5.4.23 Caution for conflicting breakpoints

[Applies to] IECUBE/ OCD(JTAG)/ OCD(Serial), V850

If there is a conflict between a software breakpoint and one of the following hardware breakpoints, the PC value may be invalidly corrected. Use a hardware break instead of a software break.

(1) Trace full break

(2) Non-map break

(3) Write-protect break

(4) Illegal I/O access break

(5) Forced break due to Stop button press

(6) Event break (hardware break)

(7) Timeout break

5.4.24 Simulating with V850E2

[Applies to] Simulator, V850E2

The instruction simulator for V850E2 supports as following functions. Other functions are not supported.
- CPU instruction
- Exceptions
- System register protection
- Memory protection
- Timing supervision function
- Floating-point operation function

Be sure to following notes.

(1) The access to external memory area is not supported

(2) A simulation result of the floating-point unit [FPU] has a margin of errors compared to real devices. The simulator uses the floating-point library of Visual C++, and store a result calculated by 80bit in a register.
(3) Following exception is not supported.
System error exception, Memory error exception
(4) The simulation of cache memory is not supported.
(5) The instructions (SYNCE/SYNCM/SYNCP) are not supported. If these were executed, the operation is same as NOP execution.
(6) The operation clock of CPU is always 4MHz. The setting of main clock on property panel is ignored.
(7) It is impossible to use data flash area. If CPU access this area, CPU breaks and error is happen.
(8) The value of Option byte storage register "OPBT0" is always "0".
(9) EH_RESET register features are not supported. In the case of a CPU reset, the reset address will always be "0x0".
(10) The number of execution clocks of each instruction will be the number of execution clocks when another instruction is executed immediately after that instruction is executed.

5.4.25 Caution on two or more variables with the same name

Applies to: All debug tools, RX

When two or more variables with the same name are defined in unnamed name spaces written in different source files, the Watch panel only shows the information on the variable that was found first.

5.4.26 Caution on member-variable pointers

Applies to: All debug tools, RX

After the member-variable pointer "mp1" defined in the program below is registered with the Watch and Local Variables panels, the type of the pointer is indicated as "int *", not "int Foo::*".

```
class Foo {
      int   m1;
};
int Foo::*mp1 = &Foo::m1;
```

5.4.27 Caution on unions assigned to registers

Applies to: All debug tools, RX

When a union is assigned to a register, it is assumed that the members of the union are assigned to the lower-order bytes of the register. For this reason, the values of the members in big endian being displayed are incorrect.

5.4.28 Caution on functions with the same name and char-type parameters

Applies to: All debug tools, RX

When three functions with char-type parameters are defined as shown below, the address of "Func(signed char)" is not displayed (i.e. the address of "Func(char)" is displayed instead).

```
void Func(char);
void Func(signed char);
void Func(unsigned char);
```

5.4.29 Caution on char-type one-dimensional arrays

Applies to: All debug tools, RX

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 or Local Variables panel even after the array "array" has been registered with the panel. (“” is shown in the column.)

```
char array[5] = "ABCD";
```

5.4.30 Caution on changing the priority section of overlay sections

Applies to: All debug tools, RX

Changing the priority section of overlay sections does not immediately reflect the debugger operation. To update the display of addresses in the editor, for example, you need to close the file and open it again. To update the display of variables in the Watch panel, single step in the program.

5.4.31 Caution on variables assigned to registers

Applies to: All debug tools, RX

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.

5.4.32 Caution on the locations where variables are assigned

Applies to: All debug tools, RX

When a defined variable satisfies both of the two conditions given below, the location of its member variables indicated in the Watch and Local Variables panels is actually the location of the entire variable.

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)
```

5.4.33 Caution on casting variables

Applies to: All debug tools, RX

In the Watch panel, class-type variables cannot be cast into base classes or derived classes. Also, structure- or union-type variables cannot be cast into any other type.

```
class AAA [
      int m_aaa;
} objA;
class BBB : public AAA {    // BBB inherits AAA.
      int m_bbb;
} objB;
class CCC {    // CCC does not inherit AAA.
      int m_ccc;
} objC

class AAA* pa = objA;
class BBB* pb = objB;
class CCC* pc = objC;
```

"(AAA*)pa"	Available
"(BBB*)pb"	Available
"(AAA*)pb"	Not available due to a restriction although class BBB inherits AAA pointed to by pb.
"(CCC*)pc"	Available
"(AAA*)pc"	Not available because class CCC does not inherit AAA pointed to by pc.

5.4.34 Caution on hardwarebreak

Applies to: OCD(JTAG) ,OCD(Serial),RX

If it uses [Go to Here] or [Execute to the specified symbol after CPU Reset],and a hardware break is set while running a program, the execution does not stop by break.

Please do not set hardware break during execution.

5.4.35 Caution on sleep state of PC

Applies to: OCD(JTAG), OCD(Serial),RX

When PC shifts to a sleep state using Windows Vista or Windows 7, Cubeuite+ cannot be continued to debug after a return from sleep state.

Please set up PC ,in order not to shift to a sleep state.

5.4.36 Caution on the trace stop and restart during program execution

Applies to: All debug tool,RX

When the trace start event or the end event set up, the trace can not stop and restart during the program execution.

5.4.37 Caution on the timestamp of trace

Applies to: OCD(JTAG), OCD(Serial),RX

When the time between frames exceeded a trace counter (20 bits), and when trace type was lost,the time stamp of trace does not display the right time.

5.4.38 Caution for Clock Generating Circuit

[Applies to] Simulator, 78K0/Kx2

The Clock Generating Circuit is simulated by the specification of 78K0Kx2.

5.4.39 Caution for Data Flash

[Applies to] Simulator, RL78 / 78K0R

The data flash is not supported.

The data flash area is treated like the usual RAM.

5.4.40 Caution for Code Flash

[Applies to] Simulator, RL78 / 78K0R

4 bytes of the final address of the code flash area and the RAM area which can be fetched cannot be fetched.

ex) When a code flash area is 0x0-0x1ffff, 0x1fffc-0x1ffff corresponds to restriction.

5.4.41 Caution for Pipeline

[Applies to] Simulator, RL78 / 78K0R

The Pipeline is not supported.

5.4.42 Flash self emulation

Applies to: IECUBE, V850

If you perform Flash self programming on the IECUBE, check whether the Flash functions shown below can be emulated, and check the related cautions.

Flash self programming Type 01

Flash Function	Functional Outline and Restriction	Availability of Emulation
FlashEnv	Flash environment initialization/end function	Emulated
FlashBlockErase	Block erasure function	Emulated
FlashWordWrite	One word writing function Restriction: If an address in the guard area is specified as the third argument, a fail-safe break occurs at an unexpected address.	Restricted
FlashBlockIVerify	Block internal verify processing function	Emulated
FlashBlockBlankCheck	Block blank check function	Emulated
FlashGetInfo	Flash information aquisition function	
	Option = 2: CPU number and total number of blocks held by CPU Restriction: The device name (four-digit number) set in the Configuration dialog box is returned as the CPU number.	Restricted
	Option = 3: Security information	Emulated
	Option = 4: Acquisition of boot area swapping information Restriction: Boot area swapping information is not reflected.	Restricted
	Option = 5 + Block number: Acquisition of last address of block	Emulated
FlashSetInfo	Flash information setting function Restriction: The boot area swapping setting is ignored.	Restricted
FlashStatusCheck	Function for checking operation status of flash function that was executed most recently Restriction: For FlashBlockErase and FlashBlockBlankCheck, the timing at which the return value changes from FE_BUSY to FE_OK differs from that in the actual device.	Restricted
FlashBootSwap	Boot area block swapping function	Not emulated
FlashSetUserHandler	User interrupt handler registration function	Emulated
FlashFLMDCheck	FLMD0 pin status check function	Emulated
FlashSetInfoEx	Flash information setting function Restriction: The boot area swapping setting is ignored.	Restricted
FlashNWordRead	Function for reading N words Restriction: If an address in the guard area is specified as the third argument, a fail-safe break occurs at an unexpected address.	Restricted

Flash self programming Type 02c

Flash Function	Functional Outline and Restriction	Availability of Emulation
FlashEnv	Flash environment initialization/end function	Emulated
FlashBlockErase	One block erasure function	Emulated
FlashWordWrite	One word writing function Restriction : If an address in the guard area is specified as the third argument, a failsafe break occurs at an unexpected address.	Restricted
FlashBlockIVerify	One block internal verify processing function	Emulated
FlashBlock-BlankCheck	One block blank check function	Emulated
FlashGetInfo	Flash information acquisition function	
	Option = 2 : CPU number and total number of blocks held by CPU Restriction : The device file name (four-digit number) is returned as the CPU number.	Restricted
	Option = 3 : Security information	Emulated
	Option = 4 : Acquisition of boot area swapping information Restriction : Boot area swapping information is not reflected.	Restricted
	Option = 5 + Block number : Acquisition of last address of block	Emulated
FlashSetInfo	Flash information setting function Restriction : The boot area swapping setting is ignored.	Restricted
FlashBootSwap	Boot area block swapping function	Not emulated
FlashFLMDCheck	FLMD0 pin status check function	Emulated
FlashWordRead	Data reading function Restriction : If an address in the guard area is specified as the third argument, a failsafe break occurs at an unexpected address.	Restricted

Flash self programming Type 03

Flash Function	Functional Outline and Restriction	Availability of Emulation
FlashEnv	Flash environment initialization/end function	Emulated
FlashBlockErase	Block erasure function	Emulated
FlashWordWrite	One word writing function Restriction: If an address in the guard area is specified as the third argument, a fail-safe break occurs at an unexpected address.	Restricted
FlashBlockIVerify	Block internal verify processing function	Emulated
FlashBlockBlankCheck	Block blank check function	Emulated
FlashGetInfo	Flash information aquisition function	
	Option = 2: CPU number and total number of blocks held by CPU Restriction: The device file name (four-digit number) is returned as the CPU number.	Restricted
	Option = 3: Security information	Emulated
	Option = 4: Acquisition of boot area swapping information Restriction: Boot area swapping information is not reflected.	Restricted
	Option = 5 + Block number: Acquisition of last address of block	Emulated
FlashSetInfo	Flash information setting function Restriction: The boot area swapping setting is ignored.	Restricted
FlashBootSwap	Boot area block swapping function	Not emulated
FlashFLMDCheck	FLMD0 pin status check function	Emulated
FlashWordRead	Data reading function Restriction: If an address in the guard area is specified as the third argument, a fail-safe break occurs at an unexpected address.	Restricted
FlashIVerify	Internal verify function (for EEPROM)	Not emulated
FlashBlankCheck	Blank check function (for EEPROM)	
EEPROM_Init	EEPROM area initialization function (for EEPROM)	
EEPROM_Write	EEPROM write function (for EEPROM)	
EEPROM_Read	EEPROM read function (for EEPROM)	
EEPROM_Copy	EEPROM copy function (for EEPROM)	
EEPROM_VChK	EEPROM valid area check function (for EEPROM)	
EEPROM_Erase	EEPROM erase function (for EEPROM)	

Flash self programming Type 04

Flash Function	Functional Outline and Restriction		Availability of Emulation
FlashInit	Self library initialization function		Emulated
FlashEnv	Flash environment initialization/end function		Emulated
FlashBlockErase	One block erasure function		Emulated
FlashWordWrite	One word writing function		Emulated
FlashBlockIVerify	One block internal verify processing function		Emulated
FlashBlockBlankCheck	One block blank check function		Emulated
FlashGetInfo	Flash information acquisition function		
	Option = 2	Device information (total number of blocks and device number)	Emulated
	Option = 3	Security flag, last block number of boot block	Emulated
	Option = 4	Device information	Emulated
	Option = 5	Reset vector address	Emulated
	Option = 6 +block number n	Last address of block number *n*	Emulated
FlashSetInfo	Flash information setting function **Restriction:** Nothing but information setting is performed. The boot area swapping setting is ignored.		Restricted
FlashStatusCheck	Checking of flash function operation that was performed last **Restriction:** SELFLIB_BUSY is not returned.		Restricted
FlashBootSwap	Boot area block swapping function **Restriction:** Functions can be called but boot swapping is not executed.		Not emulated
FlashFLMDCheck	FLMD0 pin status check function		Emulated

The cautions on performing flash self programming are described below.

No.	Description
1	Flash memory self programming emulation is not enabled in the following cases. (A) When the internal ROM size is not the default size [Workaround] Set the internal ROM size to the default value in the Configuration dialog box. (B) When using two “break before execution” [Workaround] Disable or delete one “break before execution”.
2	When flash memory self programming emulation is enabled, the following restrictions are applied to the debug function. (A) The internal ROM and internal RAM sizes cannot be changed. (B) The DMM and pseudo RRM functions are disabled. (C) An illegal break occurs in the program if the SP register value is 0 (not pointing to the internal RAM). If a break such as an event occurs before the SP register value is initialized to point to a relevant location (such as internal RAM), then it causes an illegal break for the stack area. If there is a possibility that such a break will occur during this period, set a relevant value to SP before executing the program. (D) An illegal break may occur if the restriction shown below applies to the IECUBE used. Clear the Non Map check box for the Internal RAM in the Fail-safe break setting dialog box. - An illegal break occurs during program execution in internal RAM
3	When flash memory self programming emulation is enabled, the 4-byte area starting from address 0 is reserved, a 4-byte instruction *jr 0xfffd6* is written to address 0. Therefore, when using this function at a reset vector address 0, allocate a startup routine to the area starting from address 4. If flash memory self programming emulation is disabled, 0 is written to the four bytes area starting from address 0. Do no describe codes in which execution branches to address 0, even if this function is used as a reset vector address 0. It is recommended to perform description as shown below in order to operate the same program as the one generated by emulation, in the actual device. # RESET handler (in the case of address 0) .section "RESET", text jr __start -- Overwritten by jr 0xfffd6 jr __start
4	If address 0 is specified as the reset vector handling specification address, the reset vector is set to address 4. If an address other than address 0 is specified, then the specified address is set as the reset vector without incrementing the value by four.
5	Regarding the operation of FlashStatusCheck() after FlashBlockErase() and FlashBlockBlankCheck() during emulation, the timing at which the return value of FlashStatusCheck() changes from FE_BUSY to FE_OK differs from that in the actual device.

No.	Description
6	If the address specified as the third argument of FlashWordWrite, FlashWordRead, or FlashNWordRead is located in the guard area, then an illegal memory address is accessed, and a fail-safe break occurs at an unexpected address. Correct the address to a relevant one for FlashWordWrite, FlashWordRead, or FlashNWordRead.
7	To enable the settings made in the Flash Option dialog box, be sure to reset the CPU and reexecute the program; otherwise, the setting may not take effect.
8	Secure a stack area of at least 84 (54H) bytes for the debugger's workspace. The debugger consumes a stack area of at least 84 (54H) bytes when a break occurs or during emulation processing of flash memory writing. When interrupts are enabled, a stack area of another 84 (54H) bytes is required as the debugger's workspace. If multiple interrupts are enabled, a stack area of 84 (54H) bytes must be secured per stage.
9	The data in the internal RAM is corrupted after a CPU reset. Normally, the internal RAM data after reset is not guaranteed in the actual device, but note that the operation may vary.
10	If a flash function is not used in accordance with the specifications or an unsupported flash function is called, "1" is returned.
11	The following restrictions apply to emulation of Type4. (1) An area of 48 bytes from the internal RAM end address is reserved for use by the debugger. (2) When using a device with a 1 MB internal flash memory, the internal flash area starting fromaddress 0xFF300 or higher will be used by the debugger. (3) If a flash function is executed stepwise in assemble mode, the debugger code for emulation will be executed, which is different from the code actually executed by the device. During debugging, therefore, perform stepwise execution in source mode.

5.4.43 Target system voltage during a break

Applies to: OCD(JTAG)/ OCD(Serial), 78K/RL78/V850/V850E2M

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 the debugger or communication errors.
A break during emulation of power supply off also causes communication errors.

5.5 Cautions for analysis tool

5.5.1 Cautions for Function List panel (CC-RX (C++ language))

- The following cautions apply to template functions and member functions defined in template classes.
  "(No Definition)" will appear in the "File Name" column.
  Only the types will be displayed in the Arguments column. The argument names will not be displayed.
  A hyphen ("-") will appear in the "Start Address" and "End Address" columns of member functions defined in template classes.
  If a hyphen ("-") appears in the "Start Address" column, you will not be able to jump to the Editor panel, Disassemble panel, or Memory panel.
  The Find All References menu command does not display the locations of definitions. Information about the referencing functions and variables is also not displayed.
  This feature does not count the number of function references in template functions and member functions defined in template classes. Similarly, reference information does not appear in the "Find All References" menu command.
  It is not possible to set breakpoints at the start of member functions defined in template classes from the "Set Break to Function" menu command.
- If a member function defined in a class declaration is only declared and not used, the filename will not be displayed. It will be treated as a function with no defined location.

- If you specify a function parameter with a class type, a hyphen ("-") will be displayed in the "Start Address," "End Address," and "Code Size" columns.
- If you define a function with an argument of type signed char, and an overloaded function with an argument of type char, a hyphen ("-") will be displayed in the "Start Address," "End Address," and "Code Size" columns.

5.5.2 Cautions for Variable List panel (CC-RX (C++ language))

- This feature does not display static variables defined in template functions or member functions defined in template classes.
- This feature does not count the number of variable references in template functions and member functions defined in template classes.
- The compiler changes the types of const variables without an extern/volatile declaration to constants. As a result, they will not appear in the Variable List as variables.
- Global variables with the same name defined in anonymous namespaces in different files will be treated as having the same type.

5.5.3 Cautions for Call Graph panel (CC-RX (C++ language))

- By default, template functions and member functions defined in template classes do not appear in the Call Graph panel. To display them, set the "Display the function without definition at Call Graph panel" property to "Yes."
- Functions called from/variables referenced from template functions and member functions defined in template classes do not appear in the Call Graph panel.

5.5.4 Cautions for Class/Member panel (CC-RX (C++ language))

- The following cautions apply to template functions and member functions defined in template classes.
  You cannot jump to the defined location using the Jump to Source menu command. [CC-RX]
  You cannot jump to the location of declaration in the source using the Jump to Declaration of Source menu command.
- Namespace aliases are not displayed. [CC-RX]
- The "Define Macros and Constants" node is not displayed. [CA850]
- It is not possible to jump to the type-definition location from the struct, union, or enumeration nodes. It is not possible to jump from a member node to the source code where the member is defined in the case of structs and unions. Members are not displayed for enumerations. [CX]
- If you select the member of an enumerated type and jump to the source, it will jump to the location where the enumeration is defined. [CC-RX]

5.5.5 Cautions for Variables panel

- The addresses and sizes of anonymous structs and unions cannot be displayed.
- Size information of variables that are defined only and not used will be eliminated by compiler optimization, and 0 will appear in the Size column. [CC-RX]

5.6 Cautions for Python Console

5.6.1 Caution for Japanese input

The Japanese input feature cannot be activated from the Python Console. To enter Japanese text, write it in an external editor or the like, and copy and paste it into the console.

5.6.2 Caution for prompt displays

The Python Console prompt of “>>>” may be displayed multiply, as “>>>>>>”, or results may be displayed after the “>>>”, and there may be no “>>>” prompt before the caret. If this happens, it is still possible to continue to enter functions.

5.6.3 Cautions for paths to folders and files

IronPython recognizes the backslash character (\) as a control character. For example, if a folder or file name starts with a “t”, then the sequence “\t” will be recognized as a tab character. Please use [r + "path name"] to avoid this.

Example: r"c:\test\test.py"

A forward slash (/) can be used instead of a backslash (\).

5.6.4 Caution for executing scripts for projects without load modules

If a script is specified in the startup options that uses a project without a load module file, or if project_file.py is placed in the same folder as the project file, then although the script will be executed automatically after normal project loading, it will not be executed if there is no load module file.

5.6.5 Cautions for forced termination

If the following operations are performed while a script like an infinite loop is running, then the results of function execution may be an error, because the function execution will be terminated forcibly.

1. Forcible termination by selecting “Forcibly terminate” from the context menu or pressing Ctrl+D in the Python Console
2. Changing the active project in a project with multiple projects

5.6.6 Caution for forced stops

Executing “Abort” from the context menu will forcibly stop an executing script or function, but hook and callback functions that had not started at the time the “Abort” was executed will execute in order afterward.

5.6.7 Caution on using project.Create function

If using the argument “subProject” of project.Create function, please specify only “True” or “False”.
If specify “subProject = True” or “subProject = False”, the error is occurred.

- The example that the error is occurred :
  project.Create("C:/test", MicomType.RL78, "R5F100LE", ProjectKind.Application, **subProject = True**)
- The example that the error is not occurred :
  project.Create("C:/test", MicomType.RL78, "R5F100LE", ProjectKind.Application, **True**)

5.6.8 Caution on using debugger.Watch.GetValue function (RX family)

If using “debugger.Watch.GetValue” function to read 1 byte size SFR of RX family, the read value is incorrect when the argument “watchOption” is not specified or it is specified as “watchOption = WatchOption.Auto”. So please specify watchOption except for “WatchOption.Auto”, or please use “debugger.Register.GetValue” function to read them.

Chapter 6. Restrictions

This section describes restrictions for CubeSuite+.

6.1 Restrictions for debugging tool

The below abbreviated names are used in this section.

OCD(Serial) : MINICUBE2, E1 Emulator(Serial), E20 Emulator(Serial)
OCD(JTAG) : MINICUBE, E1 Emulator (JTAG), E20 Emulator (JTAG)

6.1.1 List of restrictions for debugging tool

No.	Target tool	Target device	Description
1	OCD(JTAG)	V850E2M	Restriction for Flash options

6.1.2 Details of restrictions for debugging tool

No.1 Restriction for flash options
Applies to: OCD(JTAG), OCD(Serial), V850E2M
Description: All security settings and boot-block cluster settings of the Flash Option Settings property are invalid.
Workaround: There is no workaround

Chapter 7. Changes in User's Manual

This section describes Changes in User's Manual in CubeSuite+.

7.1 Modifications of Editor

This section describes additional functions of the editor. The editor is described in the Coding and Debugging editions.

7.1.1 Additional description of toolbar

[Before addition]
None
[After addition]

	This toggles between normal and mixed display modes when the debugging tool is connected. In normal display mode, the source program is displayed. In mixed display mode, the source program and assembly code are displayed.
	This toggles between step execution at the source and assembly level when the debugging tool is connected and in mixed display mode. At the source level, the program counter indicates the line in the source program. At the assembly level, the program counter indicates the line in the assembly code.
	This displays the current position of the program counter (PC) when the debugging tool is connected and in mixed display mode.
	This jumps to the position before executing the **Back to Last Cursor Position** command.
	This returns to the position before executing the **Jump to Funciton** command.
	This toggles between showing and hiding the column. Clicking this displays the areas that can be toggled.

7.1.2 Additional description of column headers

[Addition]

The following information items are displayed in the respective columns.

Display on the column header	Description of the items
Line	Line numbers for the files being displayed
(No display)	The display is colored to reflect the state in terms of saving of the state of editing. Yellow: The line has been newly created or modified but the file has not yet been saved. Green: The line has been newly created or modified and the file has been saved. This column is displayed except when the mixed display mode is chosen.
(No display)	The display is colored to reflect cases where a source file has been updated more recently than the corresponding load module file. The display in this area is cleared when the load module file is downloaded again after executing a build. This column is displayed if the normal display mode is chosen when a debug tool is connected.
	When the coverage function is enabled, the display is colored reflecting the results of code coverage measurement. This column is displayed when a debug tool is connected.
Address	Addresses in the microcontroller This column is displayed when a debug tool is connected.
Op code	Instruction codes This column is displayed if the mixed display mode is chosen while a debug tool is connected.
Label	Labels This column is displayed if the mixed display mode is chosen while the debug tool is connected.
	Items in this column indicate the setting of events such as timer and trace events by the context menu on lines having an address display. When an event has been set, an event mark indicating its type is displayed here.  Trace event enabled  Trace event disabled  Action event enabled  Action event disabled This column is displayed when a debug tool is connected.

	Items in this column indicate the setting of breakpoints on lines having an address display. Breakpoint settings can be made and removed through the context menu or by left-clicking on the mouse. If a breakpoint has been set, an event mark is displayed here. A mark indicating the current PC location is also displayed if the current PC location (PC register value) is included in the active panel (only in the state where the program is halted).  Breakpoint enabled  Breakpoint disabled  Current PC location This column is displayed when a debug tool is connected.

7.1.3 Additional description of dragging and dropping

[Before addition]
None
[After addition]
You can add symbols to Watch panel or Analysis Chart panel by dragging and dropping them.

7.1.4 Additional description of smart edit functions

[Addition]

The smart edit function is used to complement the names of functions, variables and the parameters of functions during input and offer them as candidates.

This function is only supported if the project is for the V850E2 or RX family. [V850E2] [RX]

The function operates with the items listed below.

- Global functions in the C language and C++ language
- Global variables in the C language and C++ language
- Class member functions in the C++ language
- Class member variables in the C++ language
- Multiple overloaded functions in the C++ language

To display candidates for functions and variables for the smart edit function, operate the program as described below.

(1) In cases where candidates are automatically to be shown

- In the C language and C++ language, if there is a relevant member for the left side when '.' is input, candidates are automatically shown at the location of the text cursor.
- In the C language and C++ language, if there is a relevant member for the left side when '->' is input, candidates are automatically shown at the location of the text cursor.
- In the C++ language, if there is a relevant member for the left side when '::' is input, candidates are automatically shown at the location of the text cursor.
- If there is a relevant method (function) on the left side of '(' when '(' is input, candidate parameter names are automatically shown.

(2) In cases where candidates are to be shown in response to keyboard and mouse operations

- When Ctrl + Space is pressed on the keyboard, all candidates are displayed at the location of the text cursor. If there is only one candidate, the relevant character string is pasted without displaying the candidate.
- When Ctrl + Shift + Space is pressed on the keyboard while the text cursor is at the location of a parameter for a method (function), a list of candidate parameter names will be shown.

- Information to complement the name of a method (function) and variable is displayed in a tooltip when a mouse cursor is placed on the partial method (function) or variable name (only when a debug tool is not connected).

Any of the actions listed below stops the display of candidates by the smart edit function.

・Pressing the Esc key

Pressing the Esc key while the smart edit function is displaying candidates for functions and variables halts the display of candidates.

・Input of a character other than an alphanumeric character while nothing is selected from the candidate list

When no candidate for a function or variable is selected, the input of a character other than an alphanumeric character halts the display of candidates.

・Input of a character other than an alphanumeric character while an item in the list of candidates is selected

When no candidate for a function or variable is selected, the input of a character other than an alphanumeric character leads to pasting of the selected candidate and thus stops the display of candidates.

Follow the procedure below to enable the smart edit function.

・In the [Text editor] page on the [Option] dialog box, select [Smart edit].

・Set the build tool property so that cross-reference information is output.

・Candidates are displayed by using cross-reference information. Execute and complete a build.

If an error in building occurs, cross-reference information before the error occurred is used if any exists (when the output of cross-reference information by the build tool is disabled, the smart edit function cannot be used since the output will be empty of cross-reference information).

Note the following items regarding the smart edit function.

・Cross-reference information is necessary.

・The smart edit function cannot be used while the mixed display mode is in use and a debugger is connected.

・The function does not support macro definitions.

・The function does not support local variables.

・The function does not support typedef statements.

・The function does not support namespaces and subclasses (classes declared within classes) of C++.

・When a structure, union or class is declared within a function in C++, candidates are not displayed within the function after its own declaration.

・The attributes const and mutable are not displayed.

・Candidates for functions and variables are not displayed in C++ in response to the input of “(*class)” or “(*this)”.

・Candidates for functions and variables are not displayed in an array declared in a class in C++.

・In C++, when candidates for functions and variables are displayed by using the Ctrl + Space key while a class name is specified to the left and input of a method name is incomplete, candidates for global functions and variables are displayed instead of candidates for functions and variables in the class.

・In some cases the type of variables to be displayed differs from that actually declared when a compiler option which affects the size of variables is set. For example: when “Yes” is set for “Replace the int type with the short type” in CC-RX, a variable declared as int is displayed as short.

・The tooltip display does not support the friend attribute of C++.

・The tooltip display does not indicate the const, static, volatile and virtual attributes.

・The tooltip display does not show structures, unions, and member functions declared in C++ header files.

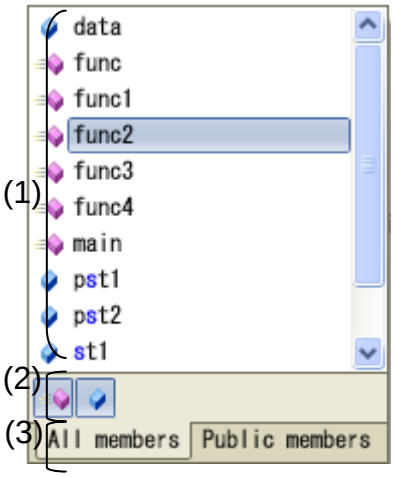
・When the build tool is CX and a union is declared, a tooltip is not displayed even if a mouse cursor is placed over the tag name of the unions.

7.1.5 Additional description of the smart editing display

[Addition]

Display of candidates for functions and variables

The smart edit function is used to display candidates for functions and variables.



The following items are described.

- [How to open the list of candidates]
- [Descriptions of the individual areas]

[How to open]

(1) In cases where candidates are automatically to be shown

- In the C language and C++ language, if there is a relevant member for the left side when '.' is input, candidates are automatically shown at the location of the text cursor.
- In the C language and C++ language, if there is a relevant member for the left side when '->' is input, candidates are automatically shown at the location of the text cursor.
- In the C++ language, if there is a relevant member for the left side when '::' is input, candidates are automatically shown at the location of the text cursor.

(2) In a case where candidates are to be shown in response to keyboard operations

- When Ctrl + Space is pressed on the keyboard, all candidates are displayed at the location of the text cursor. If there is only one candidate, the relevant character string is pasted without displaying the candidate.

[Descriptions of each area]

(1) Area for displaying candidates

Candidates for functions and variables are displayed in alphabetical order.

If a key is pressed while the smart editing display is active, the corresponding character strings are highlighted.

The following icons are displayed as labels for the list of candidates.

Icon	Description
	Shows that the candidate is for a typedef.
	Shows that the candidate is for a function.
	Shows that the candidate is for a variable.
	Shows that the candidate is for a structure or an union.

	Shows that the candidate is for a namespace.
	Shows that the candidate is for a protected member.
	Shows that the candidate is for a private member.

A candidate for a string is inserted at the location of the text cursor by selecting a candidate from the list of candidates and entering the Enter key or the Tab key.

(2) Toolbar area

Switches whether candidates for functions and variables are displayed or not.

Icon	Description
	Selected: Candidates for methods (functions) are displayed. Not selected: Candidates for methods (functions) are not displayed.
	Selected: Candidates for variables are displayed. Not selected: Candidates for variables are not displayed.

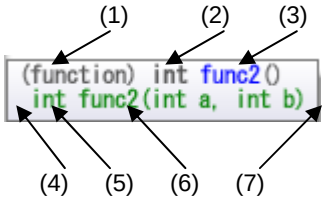
(3) Tab area

Switches the members to be displayed.

Tab name	Description
All members	All candidates are displayed.
Public members	Only the candidates with the public attribute are displayed.

Detailed display of candidates

The smart edit function is used to display details of candidates for functions and variables.



The following items are described.
- [How to open the list of candidates]
- [Descriptions of the individual areas]

[How to open]

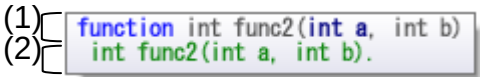
A candidate for a method (function) or variable is selected from the display of candidates for functions and variables.

[Descriptions of each area]

No.	Description	
(1)	Kind	Shows whether the selected item is a method (function) or a variable. (function): method (function) (variable): variable
(2)	Type	Shows the type of the function or variable.
(3)	Name	Shows the name of the function or variable.
(4)	Attribute	Shows the attribute (public, protected, or private). If an attribute is not defined, this item is not shown.
(5)	Type	Shows the type of the function or a variable.
(6)	Name and parameter	Shows the name of the function or a variable. When the name is a function, a parameter is also shown.
(7)	Overload information	Shows the amount of information that has been overloaded. (Example): (+1 overloads)

Display of candidates for parameters

The smart edit function is used to display details of candidates for parameters.



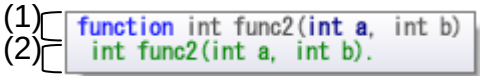
The following items are described.
- [How to open the list of candidates]
- [Descriptions of the individual areas]

[How to open]

(1) In a case where candidates are automatically to be shown
- If there is a relevant method (function) on the left side of '(' when '(' is input, candidate parameter names are automatically shown.

(2) In a case where candidates are to be shown in response to keyboard operations
- When Ctrl + Shift + Space is pressed on the keyboard while the text cursor is at the location of a parameter for a method (function), a list of candidate parameter names will be shown.

[Descriptions of each area]



No.	Description	
(1)	Candidate	Shows the name of the candidate and the parameter.
(2)	Attribute of candidate	Shows the attribute of a candidate.

When a candidate for a parameter is shown, a parameter for the location of the text cursor is highlighted (in the figure above, the first parameter is highlighted).

7.1.6 Additional description of mixed display

[Before addition]
None
[After addition]
The mixed display feature displays the source program and assembly code together; in normal display mode, only the source program is displayed.
You can set mixed display from the toolbar.
When the same file is opened again after an Editor panel that has been opened in the mixed display mode is closed, it will be displayed in the normal display mode rather than the mixed display mode. If the debugging tool is disconnected and reconnected, the file will also be displayed in the normal display mode.

Note the following when using mixed display mode.
- Editing is not available.
- Features that change a files content, such as cut and paste, delete, redo, replace, outline, and indent, are not available.
- The select all feature is not available.
- The display on the panel cannot be split.

7.1.7 Additional description of recycle mode

[Addition]

The recycle mode is used to display the sequence of multiple source files in the editor panel during debugging while the program counter (PC) enters code generated by multiple source files during stepped execution, with the multiple files of source code displayed in the same editor panel.

The recycle mode is set from [Tools] – [Options] – [Text Editor].

If an editor panel for the corresponding source file has been already opened, it will not be displayed in the panel running in recycle mode; instead, it is only displayed in the editor panel where it was already open.

Editing in the editor panel is still possible in recycle mode. When the program counter (PC) is in a given file so that file is open in the editor panel in recycle mode while a program is being executed and the contents of the panel are edited, that panel leaves recycle mode and a new editor panel is opened in recycle mode.

When an editor panel operating in the recycle mode is closed, the next editor panel to be opened will initially operate in recycle mode.

7.1.8 Change to description of split bar

[Before addition]

You can split the Editor panel display by using horizontal and vertical split bars within the view. This panel can be split up to **four times**.

[After addition]

You can split the Editor panel display by using horizontal and vertical split bars within the view. This panel can be split up to **two times vertically, and two types horizontally**.

7.1.9 Additional description of jump to functions

[Addition]

The Jump to Functions feature depends on the build tool in use and is only available when the following conditions are satisfied.

(a) When the build tool is CA78K0R, CA850, or CA78K0

· The target is a function in the active project.

· The type of project specified as the active project is “application”.

· A file with the symbol information is specified in [Download files].

However, if the file is disconnected from the debug tool, it is specified as the first file in the [Download files] property.

Note: When a file that includes symbol information is disconnected from the debug tool, jumping to static functions is not possible.

(b) When the build tool is CC-RX or CX

· When the debug tool is disconnected

- The type of project specified as the active project is "Application".
- A file with the symbol information is specified as the first file in the [Download files] property.
- The above file includes information on the target function.
- The target function is a global function.

· When the debug tool is connected to download the load module

- The downloaded load module file includes the symbol information for the function.
- Calling the target function from the file corresponding to the address of the program counter (PC).

* For example, a jump to a static function defined other than in the file corresponding to the address of the program counter (PC) is not possible.

· Furthermore, in the case of jumping to functions in C++ programs, the following point for caution also applies to specifying the functions.

- When a function a given character string is specified as the chosen function name, there is a possibility that the jump will actually be to a different function with the same name.

(1) Member functions of classes

It's necessary to include the name of the class of which the target function is a member.

When other functions have the same name as the target function but the arguments are different, please also include the argument type.

Example: "memfunc": Jumping is not possible.

"Class::memfunc(short)": Jumping is possible.

(2) Functions defined in Namespaces

It's necessary to include the full name of the Namespace to which the target function target belongs.

When other functions have the same name as the target function but the arguments are different, please also include the argument type.

Example: "func": Jumping is not possible.

"Namespace1::Namespace2::func(int)": Jumping is possible.

(3) Template functions

It's necessary to include the type of the arguments in the function generated by the compiler.

Example: "template": Jumping is not possible.

"template(int, short)": Jumping is possible.

(c) External build tools

· The target is a function in the active project.

· A file with the symbol information is specified in [Download files].

However, if the file is disconnected from the debug tool, it is specified as the first file in the [Download files] property.

Note: When a file that includes symbol information is disconnected from the debug tool, jumping to static functions is not possible.

7.1.10 Additional description of the context menu

[Addition]

ProgramAnalyzerAddVariableToChart	Registers a variable in the [Variable Value Changing Chart] tab of the Analysis Chart.

7.1.11 Additional description of the Print Preview dialog box

[Before addition]

(1) Preview area

The image to be printed is displayed as a preview.

[After addition]

(1) Preview area

The image to be printed is displayed as a preview. The display differs according to whether the debug tool is or is not connected, and when it is connected, to whether the display is in normal mode or mixed mode.

When the outline setting is in used and a collapsed section of an outline is displayed in a print preview, the lines in the collapsed section are also displayed.

(a) When the debug tool is not connected

The display is as follows.

Left of page: line number

However, this is not displayed when line numbers are hidden on the editor panel.

Page header: file name (fully qualified path)

Page footer: page number

(b) When the debug tool is connected (in normal display mode)

The display is as follows.

Left of page: line number and address

However, these are not displayed if line numbers and addresses are hidden on the editor panel.

Page header: file name (fully qualified path)

Page footer: page number

(c) When the debug tool is connected (in mixed display mode)
The display is as follows.
Left of page: line number, address, instruction code
However, any of these items that are hidden on the editor panel are not displayed.
Page header: file name (fully qualified path)
Page footer: page number

7.2 Modifications in Start

This section describes modifications in CubeSuite+ V1.02.00 Start User's Manual (document # R20UT0975EJ0100).

7.2.1 Additional description of the [General – Text Editor] category of the Option dialog box

[Location] Page 228
[Before addition]
This option is used to encode the default file which is required when a new file is created in the editor panel.
[After addition]
This option is used to encode the default file which is required when a new file is created in the editor panel and automatic determination of encoding is disabled.

7.2.2 Additional description of the [General – Text Editor] category of the Option dialog box

[Location] Page 228
[Addition]
(e) Smart editing
(checked)
The smart edit function is enabled (default).
(not checked)
The smart edit function is disabled.
(f) Smart editing
(checked)
When a file is read, automatic determination of the encoding is enabled (default).
(not checked)
When a file is read, automatic determination of the encoding is disabled.

7.2.3 Change description of the [General – Text Editor] category of the Option dialog box

[Location] Page 228
[Before addition]

☑	Uses window recyvling (default).
☐	Does not use window recyvling.

[After addition]

☑	Uses window recyvling.
☐	Does not use window recyvling**(default)**.

7.3 Modifications in Build

This section describes modifications in CubeSuite+ V1.01.00 Build User's Manual (document # R20UT0730EJ0100, R20UT0783EJ0100).

7.3.1 Additional description of the stack usage tracer

[Location] Page 315 → Functions analyzed

[After addition] Consequently, functions in assembly files written by the user and library files created by the user are not analyzed. For this reason, the information for these files must be set using the Adjust Stack Size dialog box.

And, since an interruption function also becomes the outside of analysis management, it is necessary to set up applicable information using a adjust stack size dialog box.

[Location] Page 324 → Functions analyzed

[After addition] Consequently, functions in assembly files written by the user and library files created by the user are not analyzed. For this reason, the information for these files must be set using the Adjust Stack Size dialog box.

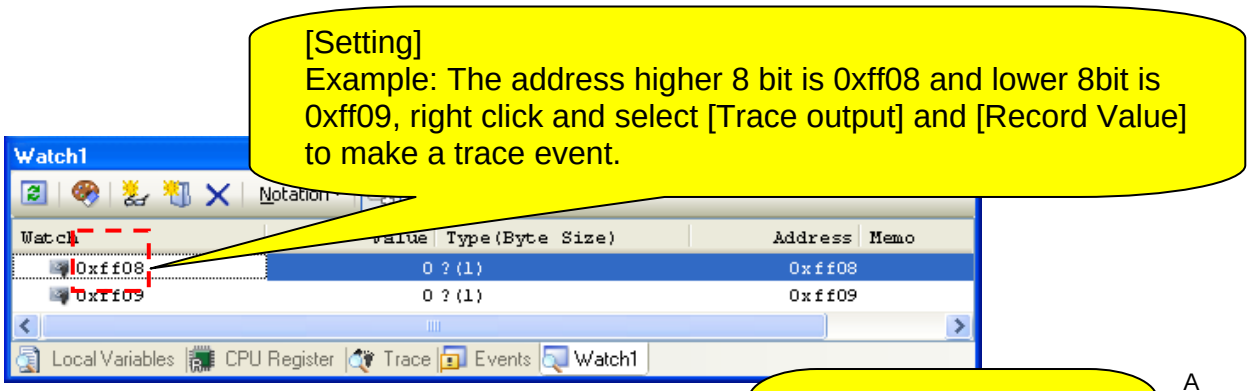
And, since an interruption function also becomes the outside of analysis management, it is necessary to set up applicable information using a adjust stack size dialog box.

7.4 Modification in 78K0 debug

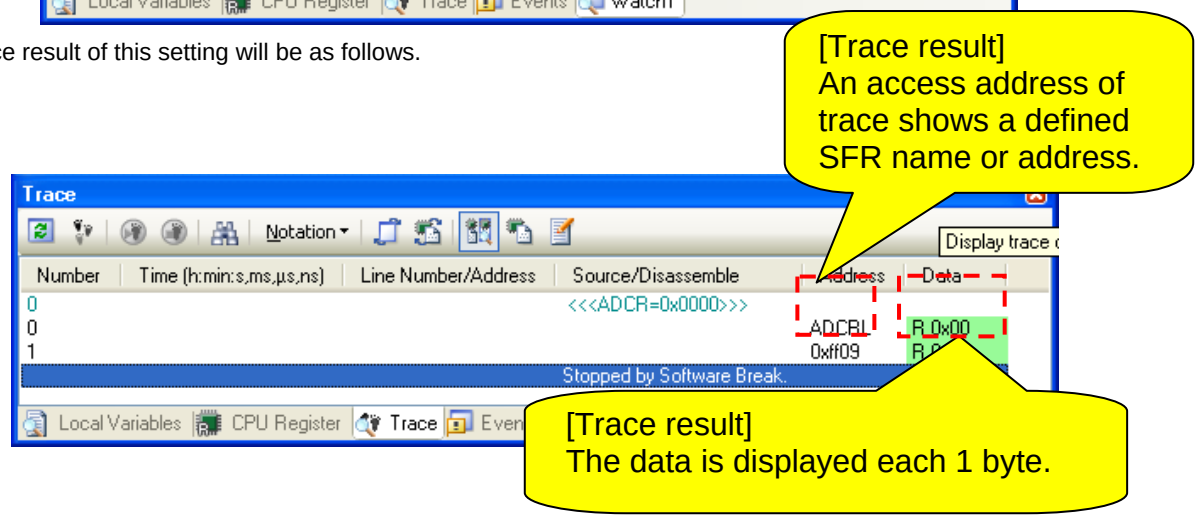
This section describes modification in CubeSuite+ V1.01.00 78K0 Debug User's Manual (document# R20UT0731EJ0100).

7.4.1 Addition of point trace description Special Function Registers (SFRs) /variables of 2 byte size.

[Location] page 118 Addition before 「2.11.5 Display the collected execution history」
[Addition]
2.11.4(2) When an access to SFRs/variables of 2byte size occurs. [IECUBE]
Register each address of the higher 8 bit and the lower 8 bit in watch panel directly and make a trace event to do point trace of SFRs/variables of 2 byte size. (See the figure below)



A trace result of this setting will be as follows.



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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

RENESAS

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

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

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

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

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

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

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

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

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

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

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

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

Colophon 2.2