### RX development environment migration guide

Migration From RX Family and Other Family to RX Family

(IDE ed.)

(High-performance Embedded Workshop to CS+)

November 30, 2016 R20UT2055EJ0102

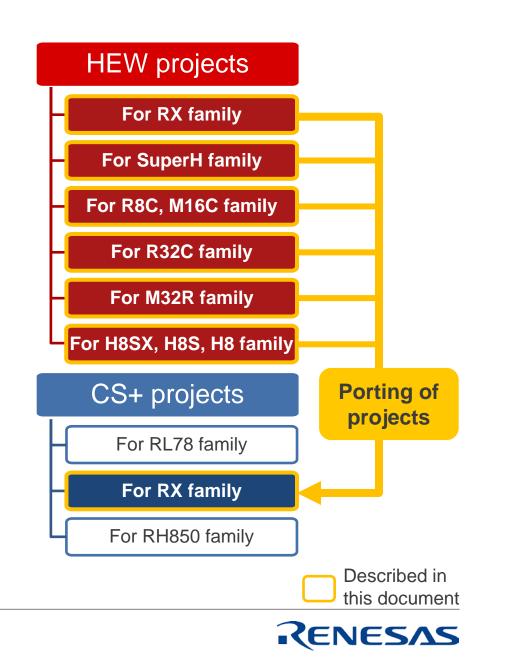
Software Product Marketing Department, Software Business Division Renesas System Design Co., Ltd.



#### Preface

 This document describes the porting a project from Highperformance Embedded Workshop (hereinafter HEW) IDE to CS+ IDE for an RX family compiler (hereinafter CC-RX).

- The applicable versions of CS+ / HEW / CC-RX are as follows.
  - CS+ for CC V4.01.00
  - HEW V.4.09.01.007
  - CC-RX V2.05.00



# Agenda

<ul> <li>Utilizing Existing Projects</li> </ul>	Page 4
<ul> <li>Differences from HEW when Creating a Project</li> </ul>	Page 5
Migration to CS+ for RX from HEW for Non-RX	Page 18



## **Utilizing Existing Projects**

Workspaces\* created by the HEW can be loaded for use in the CS+ environment.



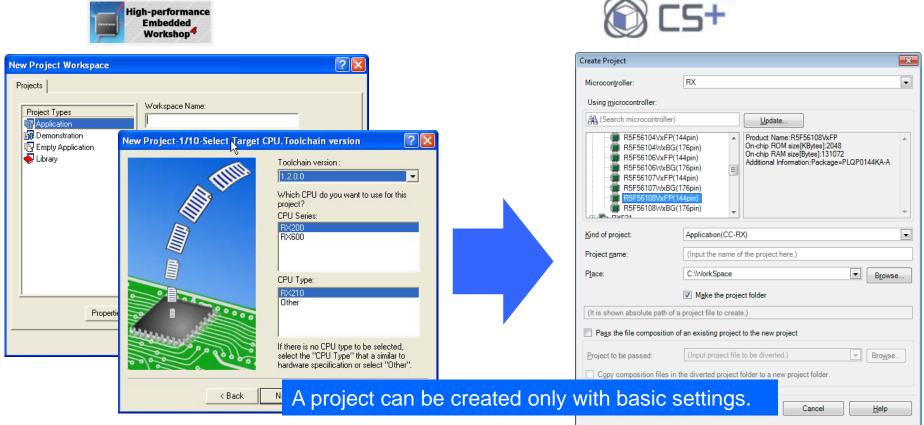
Start		- x	Opened in the CS+ environment
	GO We recommend reading the tutorial to find out what can be done in CS+. The tutorial contains the information on how to effectively use CS+.		taking over the following settings:
C	A new project A new project can be created. A new project can also be created by reusing the file configuration registered to an existing project.	-	<ul><li>Files</li><li>Project configuration</li></ul>
C	create New Multi-core Project	+	.,
0	Den Existing Project Loads the project of CS+. Can also be opened directly from the following link.	-	<ul> <li>Build options (only from HEW for RX</li> </ul>
	GO Recent Projects Favorite Projects 1. RX610_Tutorial_DebugConsole Nothing		
-0	Open Existing e² studio/CubeSuite/High-performance Embedded Workshop/PM+ Project         The project created with e² studio and the old IDE can be converted to the CS+ project.         Support version:         e² studio         The rcpc file output by e² studio can be read.         (I) Build options also can be converted between the projects with the same compiler (Only CC-RX is supported in this version).         (I) Only include path and macro options can be converted between the projects with the different compiler.	-	* Workspaces for RX created by HEW v.4.07 or later
-0	pen Sample Project		

Migration from HEW to CS+ is supported.



#### Differences from HEW when Creating a Project(1/8)

• The setup wizard for creating a project is different.





#### Differences from HEW when Creating a Project(2/8)

 Settings made in the process of creating a project in the HEW can be made in the window shown below after a project has been created.



Make settings in the [Common Options] page of CC-RX (build tool) properties after a project has been created.

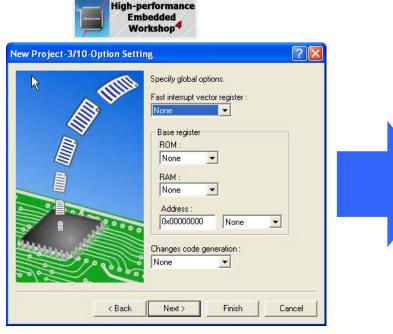


Build mode	DefaultBuild	
Ghasge-property value for all build modes at once	Na	
CPU		
Instruction set architecture	RXv1 architecture(-isa=rxv1)	
Uses floating-point operation instructions	Yes(fpu)	
Endian type for data	Little-endian data(-endian=little)	
Rounding method for floating-point constant operations	round to nearest(-round=nearest)	
Handling of denormalized numbers in floating-point constants	Handles as zeros(-denormalize=off)	
Precision of the double type and long double type	Handles in single precision(-dbl_size=4)	
Replaces the int type with the short type	No	
Sign of the char type	Handles as unsigned char(-unsigned_char)	
Sign of the bit-field type	Handles as unsigned(-unsigned_bitfield)	
Selects the enumeration type size automatically	No	
Order of bit-field members	Allocates from right(-bit_order=right)	
Assumes the boundary alignment value for structure members is 1	No(-unpack)	
Enables C++ exceptional handling function (try, catch and throw)	No(-noexception)	
Enables the C++ exceptional handling function (dynamic_cast and typeid)	No(-tti=off)	
General registers used only in fast interrupt functions	None(-fint_register=0)	
Branch width size	Compiles within 24 bits(-branch=24)	
Base register for ROM	None	
Base register for RAM	None	
Address value of base register that sets the address value	HEN 0000000	
Register of base register that sets the address value	None	
Avoids a problem specific to the CPU type	Yes(for RX610 Group)(-patch=rx610)	
Saves and restores ACC using the interrupt function	No	



#### Differences from HEW when Creating a Project(3/8)

 Settings made in the process of creating a project in the HEW can be made in the window shown below after a project has been created.



Make settings in the [Common Options] page of CC-RX (build tool) properties after a project has been created.

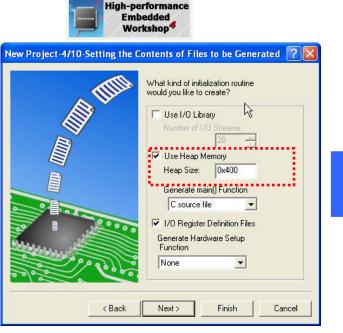


~	Property	
4	CC-RX Property	
۵	Build Mode	
	Build mode	DefaultBuild
4	Change property value for all build modes at once CPU	No
	Instruction set architecture	RXv1 architecture(-isa=rxv1)
	Uses floating-point operation instructions	Yes(fpu)
	Endian type for data	Little-endian data(-endian=little)
	Rounding method for floating-point constant operations	round to nearest(-round=nearest)
	Handling of denormalized numbers in floating-point constants	Handles as zeros(-denormalize=off)
	Precision of the double type and long double type	Handles in single precision(-dbl_size=4)
	Replaces the int type with the short type	No
	Sign of the char type	Handles as unsigned char(-unsigned_char)
	Sign of the bit-field type	Handles as unsigned(-unsigned_bitfield)
	Selects the enumeration type size automatically	No
	Order of bit-field members	Allocates from right(-bit_order=right)
	Assumes the boundary alignment value for structure members is 1	No(-unpack)
	Enables C++ exceptional handling function (try, catch and throw)	No(-noexception)
	Enables the C++ exceptional handling function (dynamic_cast and typeid)	No(ttti=off)
	General registers used only in fast interrupt functions	None(-fint_register=0)
	Branch width size	Compiles within 24 bits(-branch=24)
	Base register for ROM	None
	Base register for RAM	None
	Address value of base register that sets the address value	HEX 0000000
	Register of base register that sets the address value	None
	Avoids a problem specific to the CPU type	Yes(for RX610 Group)(-patch=rx610)
	Saves and restores ACC using the interrupt function	No



#### Differences from HEW when Creating a Project(4/8)

 Settings made in the process of creating a project in the HEW can be made in the window shown below after a project has been created.



In the HEW, the set value is applied to the generated startup code.



sbrk.h file

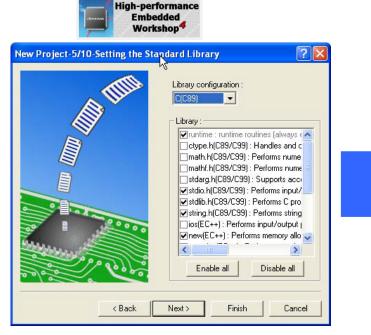
Property Sbrk.h
🚯   🖏   ⇒ 😋 🗠   Columns▼
L 🕝 1 /* size of area managed by sbrk */ 2 #define HEAPSIZE 0x400
3

Setting can be updated by changing the value.



#### Differences from HEW when Creating a Project(5/8)

 Settings made in the process of creating a project in the HEW can be made in the window shown below after a project has been created.



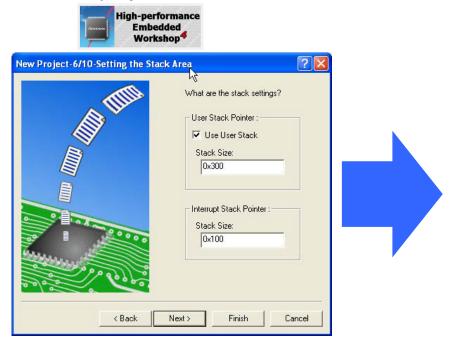
Make settings in the [Library Generate Options] page of CC-RX (build tool) properties after a project has been created.



8	CC-RX Property Mode		_ <b>^</b> _
•	Generation mode of the standard library Standard Library	Build a library file (option changed)	
	Library configuration	C(C89)(-lang=c)	
	Configuration library	Custom(-head= <suboption>)</suboption>	
	Enables runtime library	Yes(-head=runtime)	
	Enables ctype.h(C89/C99)	Ne	
	Enables math.h(C89/C99)	No	
	Enables mathf.h(C89/C99)	No	
	Enables stdarg.h(C89/C99)	No	
	Enables stdio.h(C89/C99)	Yes(-head=stdio)	
	Enables stdlib.h(C89/C99)	Yes(-head=stdlib)	
	Enables string.h(C89/C99)	Yes(-head=string)	
	Enables ios(EC++)	No	
	Enables new(EC++)	Yes(-head=new)	
	Enables complex(EC++)	No	
	Enables string(EC++)	No	
	Object		
	Path of the output folder	%BuildModeName%	
	Output file name	%ProjectName%.lib	
	Creates a functional cutdown version of I/O functions	No	
	Creates the reentrant library	No	
	Check memory smashing on releasing memory	No	
	Use same object-related settings as Compile Options tab	No	
	Section name of program area	P	
	Section name of constant area	С	

#### Differences from HEW when Creating a Project(6/8)

 Settings made in the process of creating a project in the HEW can be made in the window shown below after a project has been created.





Described on the next page



#### Differences from HEW when Creating a Project(7/8)

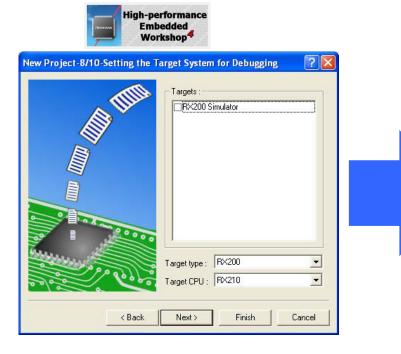
 RX's stack areas are automatically allocated according to the description in the stacksct.h file and the stack section layout set on the [Link Options] page of CC-RX (build tool) properties.

	<stacksct.h> #pragma stacksize su=0x100 #pragma stacksize si=0x300</stacksct.h>	✓ Section Section ✓ The s	ization type Oni on start address pecified section that outputs externally defined symbols to the file	B_1,R_1,B_2 The specified	section that out	//04,PResetPRG/0FFE000 puts <del>*externally</del> defined syn		\$DSEC,C\$BSEC,C\$INIT,C\$VTBL,C	c
Stack sizes written in the stacksct.h file		<ul> <li>▷ ROM</li> <li>▷ Verify</li> </ul>	<ul> <li>▷ Verify</li> <li>▷ Others</li> </ul>		ment[0] M mapped section[3] Section Settings				
		Commo	on Options 🔏 Compile Options 🔏 Assemble Options 🔪 <b>Link Options</b> 🔏 Hex	Output Op	Address 0x00000004	Section           B_1           R_1           B_2           R_2           R_2		Add           Modify           New Qverlay           Remove	•
	••• MVTC #00000508,USP		Stack section layout setting		0x0FFE00000			Lp Down	
Autoi	MVTC #00000808,ISP	uild	Generates a code in which the address out from "addresses at which SU and S are allocated" + "sizes specified in the s file" is specified as the initial value of the pointer.	l secti tackso	ons ct.h	С_2	Cancel	★ Export	

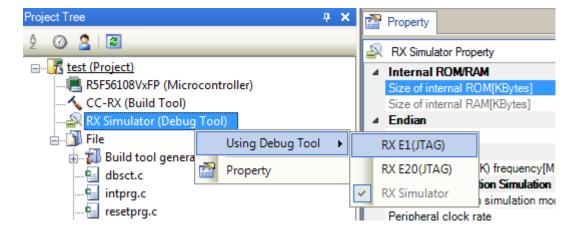


### Differences from HEW when Creating a Project(8/8)

 Settings made in the process of creating a project in the HEW can be made in the window shown below after a project has been created.



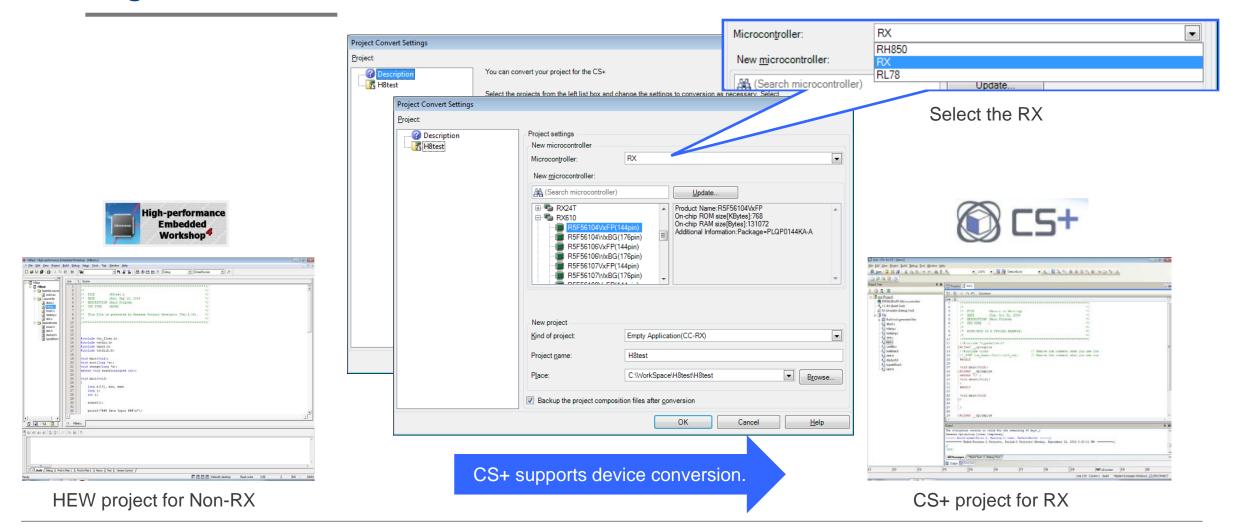




In the CS+ environment, created projects can be easily connected to the debugger environment.



### Migration to CS+ for RX from HEW for Non-RX





#### Renesas System Design Co., Ltd.

